Towards Individualized Production with Flexible Shopfloor IT based on Distributed OPC UA Services

Tobias Wolff, Axel Vick
Fraunhofer Institute for Production Systems and Design Technology, Pascalstraße 8-9, Berlin, GER
E-mail: axel.vick@ipk.fraunhofer.de

Abstract. Nowadays, manufacturers face the problems to adjust their production according to the fast-changing global competition. Car manufacturers, for example, are in front of a new era of individual mobility. They need to switch from combustion engines to electric drives. In general, manufacturers have to develop production strategies to lower set up time and meet requirements faster. The classical production line is most effective if the product variety is low and the lot size is high but facing the rapid growing demand for individualized products and fast-changing requirements, manufacturers must discover new ways of manufacturing. One possibility is the flexible shopfloor. In this work, we will present our concept of a flexible shopfloor IT for controlling a robotic demonstrator. We use OPC UA over TSN as a platform for communication and CODESYS as SoftSPS for the control of sensors and actuators. In order to create valid data for ongoing discussions we use function point analysis and counting lines of sourcecode to measure software size. Our expectation is that the initial effort for implementing a flexible shopfloor IT is higher compared to the classic approach, but for all the following implementation it will take less time, because of the possibility of reuse already developed systems.

1. Introduction

The topic of flexiblity within a company is research topic since the 1980s. In [1], Grob discussed flexiblity with a broad view and named different variation of flexiblity like flexiblity of employees, of process and organisation structure and flexiblity of production systems. Within the manufacturing flexiblity can be splitted into more appropiate definitions like batch size, variations of products and production concepts. In this paper we focus on the flexible production systems which is one concept of production. In [1] Grob named reasons why companies are forced to increase their flexiblity within their organisation.

The key factors are:
• Rapid changing Global Market,
• Changing Technology,
• Laws and International Regulations,
• Mindset of Society regarding Labor.

For each key factor the recent history provides examples like climate change; rethinking individual mobility; news are distributed faster and trends arise faster and stronger; and young people with different ideals. Companies face those different challenges and much more in the nearest future. They have to find new ways to increase their flexibility.

Concluding the chapter we can formulate the research question: Does a flexible production reduce the time needed, when reconfiguration is required?

2. Related Work

Industry 4.0 and digitization are intended to find solutions to the problems mentioned. With OPC UA, MQTT, CoAP technologies and standards have been developed that enable stronger networking. OPC UA is seen here as the de facto standard for Industry 4.0, since it enables vertical and horizontal networking of systems. With Time-Sensitive Networking (TSN), OPC UA also becomes real-time capable and thus time-critical applications can be controlled with OPC UA. One example is the control of robots. Communication in OPC UA is based on a client/server principle. This can be a disadvantage in complex systems, since every connection must be actively maintained from both sides. The OPC Foundation extends the standard by the Publish/Subscribe principle. Subscribers create a subscription to a topic and then receive a notification in case of a change or a defined time interval. With [10] Pfrommer et al. present a flexibly structured manufacturing IT. They use OPC UA to network their developed systems - which are on different levels in the automation pyramid - with each other. Pfrommer et al. use AutomationML to describe the capabilities offered by the system. The system sends this description to the central controller when it logs into the network. In [3] Jaekel et al. present their concept of flexible manufacturing, the modular shop-floor IT. The core elements here are the process definition using Integrated Enterprise Modeling (IUM), the Execution Engine [4] and the Cyber Physical System Adapter (CPS Adapter). With IUM, processes and the required machines, resources and products can be defined. The Execution Engine controls the machines and is the link between the IUM and the machines. The CPS Adapter is responsible for connecting the machines to the Execution Engine and thus to the higher-level IT. It forms an intermediate layer and enables machines without OPC UA to be integrated into an IT infrastructure based on OPC UA. Older machines can thus continue to exist in Industry 4.0. With TSN a real time communication based on Ethernet is developed. In [7], [9], [10], and [6] it is presented in combination with OPC UA to enable real-time capable and distributed control.

3. Methods and Material

The work presented in chapter 2 is extended by some points. In figure 1 you can see our architecture and our idea of a flexible production system. Instead of the idea of the CPS Adapter we use CODESYS as a link between machine and IT infrastructure. This has the advantage that a programmable logic controller (PLC) is redundant and the control of the sensors and actuators is done directly from a software. The CODESYS instance is therefore hardware independent and in combination with TSN can run physically far away from the machine in a container - in our case, a Linux server. The robot in Figure 1 is controlled this way. We also follow the edge computing approach. In complex systems the centralization of computing capacities is critical. The idea of edge computing is to distribute the computing resources and bring them closer to
the source of the data, so that processing of the data is done at the source and only the derived data is sent to the central controller.

On the left side of Figure 1 the levels of the automation pyramid are illustrated so that the system can be categorized by their purpose. With OPC UA this hierarchy is redundant because systems can be directly connected to each other. The core system is the Workflow-Engine as centralized control system. We outsourced some functionalities into new systems to meet the Edge-Computing paradigm. The ServiceLib and the Local Discovery Server (LDS) are two of them. The Workflow-Engine is responsible for executing processes and creating service descriptions of newly added services. A web service description language file is created for every service. After creation the Workflow-Engine calls an OPC UA method of the serviceLib which saves the file. Those files are needed when a user defines a new process in the graphical user interface (GUI). To be as flexible as possible we used a LDS [11]. Every newly added OPC UA server registers itself at the discovery server. Every client can use the OPC UA service `FindServers` to get a list of available OPC UA server within the system.

To create data on which we can discuss our approach and find an answer on our research question we need quantization. Our development is software only and that is why we use software estimation methods to analyze it. In [13] the first method of software sizing was introduced and this approach is still used. It is based on counting lines of source code. This counting is good for software at the development phase or for finished products. Another well known approach is the Function Point Analysis [14]. It is standardized by the International Function Point User Group (IFPUG) and was recently recommend by the European Parliament for price calculations.
of software based on its size [15]. Function Point Analysis is used at the early stages of projects where only the architecture exists. In our architecture we use different programming languages like C++ for the Workflow Engine, Java for the GUI and structured text for the PLCs. The function point analysis is best suitable for comparing systems programmed in different languages. Indeed it is not applicable for PLC programming. A program of a PLC is executed in cycles whereas in C++ or Java works sequential. In one cycle the PLC reads all inputs, executes all subprograms and writes the outputs. The function point analysis relies on functions, which do not exist in PLC programming. Furthermore function point analysis focuses on the written code and not on the compiled program. Reading and writing inputs and outputs in a cycle is an analog digital conversion and not manipulating data. That is why we chose counting lines of source code.

4. Experiments
To prove or disprove our statement we did three experiments. At first we defined a process and applied it to our three use cases. Those three scenarios are our three experiments. Our first case is the classic approach where we develop a PLC program. We need this reference value in order to compare our other results. The second experiment is our initial implementation of our concept. In our third experiment we replace the services with C++ programs. Those programs simulate a cyber physical system. Replacing services in our architecture is our kind of reconfiguration.

5. Results and Discussion
Our exception was, that a flexible organized production reduces the time needed if reconfiguration is needed. We cannot confirm that. The amount of lines of the classic approach is nearly the same compared to flexible production. The reason for this is the increased amount of functionalities within our architecture. The distributed system based on OPC UA and Cloud Computing results also in a vertical integration within the automation pyramid. The implementation of OPC UA needs a lot development effort which leads to an increased amount of source code. Critical is that the amount of source code over time is not linear. We gained a lot experience of OPC UA an open62541 as C++ library so that we can develop the same application faster than at the beginning of this project. A better method for quantization is needed for our use case. We welcome the further development of the function point analysis. In the current version it offers the best solution to compare software programmed in different languages, but it lacks by analysing a PLC program.

6. Conclusion
The work presented in this paper has two outcomes. The first one is a combination of different technologies to create a flexible production system. Here, the edge-computing paradigm took a place and we showed with CODESYS, OPC UA and TSN how we achieved this. CODESYS makes the hardware programmable logic controller redundant and can run on different systems like servers or computers with different operating systems. CODESYS implements the Micro Embedded Device Server which summarizes functionalities and is therefore not able to implement methods or Publish/Subscribe [12]. A future implementation of both will make CODESYS even more suitable. OPC UA as promising standard is implemented and supports the edge-computing concept with the LDS and the platform for connecting several devices from different vendors from different levels of the automation pyramid. TSN makes real-time critical applications possible. Especially in Level 1 and Level 2 of the classic automation pyramid the time delay has to be within a few milliseconds with a few microseconds of jitter to meet PLC take cycle times requirements.
We neglected the definition of processes in our layout. Jaekel et al. [3] and Torka [4] contributed a lot of research here. The implementation of a process definition tool into our architecture would increase the flexibility of our proposed production system. Furthermore, the integration of an enterprise ressource planning system would overcome the last remaining layer of the automation pyramid. Another part which is out of focus of this paper is the time needed for reassembling the hardware, organization and supply chain management.

The second outcome is the analysis of developing effort of such a system. With the lines of code and Function Points we used two different methods for software size measurement. Our initial implementation takes more time than the classical approach by directly programming PLCs. We could not show that our system reduces the effort needed for configuration, because our proposed architecture has more functionalities than implementing a process directly in a PLC. Besides this analysis we discovered the problem of appropriate methods for quantization. We showed that the combination of edge-computing with OPC UA over TSN and CODESYS as software PLC is one concept of many flexible production systems concepts which was one of our goals. The second contribution of this paper is our analysis, where we tried to show that a flexible production saves development time. Here we showed that measuring software size is still a challenge. More research has to be done on both: flexible production systems and appropriate methods for quantization.

7. References

[1] Grob R 1986 Flexibilität in der Fertigung (Berlin, Heidelberg: Springer) p 2 ff.
[2] Wintrich N, Knothe T, Eggers A, Riedel O, Stölze S, Margraf T 2014 Modellbasierte modulare Shopfloor IT - Integration in die Werkzeuge der Digitalen Fabrik (Germany Berlin: Fraunhofer IPK)
[3] Frank-Walter J, Torka J, Eppelein M, Schliephack W, Knothe T 2017 Model Based, Modular Configuration of Cyber Physical Systems for the Information Management on Shop-Floor (Greece Rhodos: On the Move to Meaningful Internet Systems (OTM))
[4] Torka J 2017 Modellbasierte Execution Engine für die Ablauflogik der modularen Shopfloor IT in der Produktionstechnik (Germany Berlin: Beuth Hochschule für Technik Berlin)
[5] Saracco R 2017 Carrying 10 Exabyte of cars generated data
[6] Zhou Z, Shou G 2019 An Efficient Configuration Scheme of OPC UA TSN in Industrial Internet (China Hangzhou: Chinese Automation Congress (CAC))
[7] Gogolev A, Mendoza F, Braun R 2018 TSN-Enabled OPC UA in Field Devices (Italy Turin: 2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA))
[8] Eckhardt A, Müller 2019 Analysis of the Round Trip Time of OPC UA and TSN based Peer-to-Peer Communication (Spain Zaragoza: 2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA))
[9] Eymüller C, Hanke J, Hoffmann A, Kugelmann M, Wolfgang R 2020 Real-time capable OPC-UA Programs over TSN for distributed industrial control (Austria Vienna: 2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA))
[10] Pfrommer J, Stogl D, Kiril A, Schubert V, Hein B 2014 Real-time capable OPC-UA Programs over TSN for distributed industrial control (Spain Barcelona: 2014 IEEE International Conference on Emerging Technology and Factory Automation (ETFA))
[11] OPC Foundation 2018 OPC Unified Architecture Part 12: Discovery and Global Services Release 1.04
[12] OPC Foundation 2018 OPC Unified Architecture Part 7: Profiles Release 1.04
[13] Boehm B, Clark B, Horowitz E, Westland C, Madachy R, Selby R 1995 Cost models for future software life cycle processes: COCOMO 2.0 (Annals of Software Engineering volume 1) p 57 ff.
[14] Albrecht A, Gaffney J 1983 Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation (IEEE Transactions on Software Engineering Volume SE 9) p 639 ff.
[15] European Parliament 2020 European Union Agency for the Operational Management of Large Scale IT Systems in the Area of Freedom, Security and Justice (eu LISA) (Belgium Brussel)