Preprints of the 18th IFAC World Congress
Milano (Italy) August 28 - September 2, 2011

Leveraging the Agility of Manufacturing Chains by Combining Process-Oriented Production Planning and Service-Oriented Manufacturing Automation

Lisa Ollinger*, Jochen Schlick, Stefan Hodek****

**Institute for Production Automation, University of Kaiserslautern, Kaiserslautern, Germany (e-mail: lisa.ollinger@mv.uni-kl.de)}*
***Innovative Factory Systems, German Research Center for Artificial Intelligence Kaiserslautern, Germany (e-mail:{ jochen.schlick; stefan.hodek}@dfki.de).*

Abstract:

Globalization, rising product variety, the need for continuously improving productivity and quality demand higher agility of manufacturing systems. Well-known principles like encapsulating functionality into mechatronic systems to enable reuse and standardization are basic approaches to gain agility in manufacturing environments. However, there is a lack of a holistic communication architecture for manufacturing systems and an overall concept to organize the continuously changing production planning and control processes. The paradigm of service-oriented architectures emerged as a concept to increase the flexibility and reuse within IT environments by using software modules with standardized communication interfaces. Transferring this paradigm to the field of manufacturing offers a unique opportunity to complement the advances of standardized communication interfaces and mechatronic encapsulation with powerful production planning and control methods.

In this paper the central aspects and the potentials of transferring service-oriented paradigms from IT to automation are discussed. The methodology of process-oriented manufacturing planning is presented as the organizational fundament for an efficient establishment of service-oriented manufacturing systems. An approach for a process-oriented factory model is presented as the basis for a process-oriented planning process. Furthermore, a technical demonstrator that provides the opportunity for evaluating SOA technologies and our new planning approach is shown.

Keywords: **Manufacturing control, Flexible manufacturing systems, Production control, Production systems, Agile Manufacturing, Control systems**

1. INTRODUCTION

The economy of the 21st century is characterized by global competition, shortening innovation and product lifecycles and an increasing customer demand for highly individualized goods. In order to stay competitive under such conditions, manufacturers need to make their factory systems more flexible and versatile for being able to quickly adapt to changing market demands.

Typically changes in products as well as new products that are related to old ones lead to frequent reconfigurations of existing manufacturing lines. This means that producers, engineering companies, and plant manufacturers have to adapt production equipment in the shortest possible time and with minimal effort to new requirements and boundary conditions. Therefore, modularization and reuse of equipment are central paradigms in today's factories. Up to now, one of the main difficulties when reconfiguring equipment is the integration of modules in existing systems. Often, integrating modules means writing glue code to adapt interfaces of different standards to each other.

Today, modularization of process steps or automation modules is supported by recent technical developments in automation which accompany the principle of encapsulation of mechanic functions. Abstraction and standardization of field bus protocols and the increasing computational power of field bus components support the modularization of control systems in equipment and plants. The advancement of specialized sensor-actor protocols to standard field bus protocols like Industrial Ethernet minimizes the variation of industrial digital communication systems. From the engineering point of view this in turn decreases the integration effort. Fewer protocol interfaces will have to be used and the experience with the standard bus protocol speeds up integration work. Many existing field bus components already have the ability to take over simple control tasks e.g. by running a small PLC. This promotes decentralized automation so that distributed control gets more and more important in the field. Each sub process can be automated with its own control using a dedicated hardware. This in turn leads to encapsulated functional modules realizing one or more sub processes [Vogel-Heuser *et al*. 2009].

In order to combine those functional modules to complete production systems the aspect of interoperability gets more and more important. The effort to integrate functional modules into the production line, to connect the control

system to manufacturing execution systems and enterprise resource planning systems strongly depends on adequate interfaces and data models. Therefore, standardization issues and the introduction of middleware are typical approaches to increase the interoperability of factory automation systems.
Continuing the idea of middleware the paradigm of service-oriented architectures (SOA) enhances interoperability of IT systems in a much more flexible way. In order to establish communication bridges between different pieces of software no middleware has to be written. The basic idea of service-oriented architectures is that all involved software modules – which are called services – are self describing and can be integrated in an interoperable way. The self descriptiveness bases on a model based company wide description of communication and information models.
Until now, some approaches have already been developed that dealt with the set-up of service-oriented architectures in factory and manufacturing automation. The two outstanding approaches arise from the EU founded research projects SOCRADES and PABADIS'PROMISE. In PABADIS' PROMISE a control architecture was developed that enables a decision responsibility distribution among acting control entities [PABADIS'PROMISE Consortium 2008]. The ERP system comprises several classes of business functions that are encapsulated as services. The connection to the agent-based MES system is realized through web services. The SOCRADES consortium developed a web service based communication architecture for the industrial automation domain [Souza *et al.* 2008]. The focus was the connection of field devices with high-level control systems like MES and ERP systems. Therefore, the communication technologies DPWS and OPC-UA were used.
The state of the art of the application of SOA in the field of industrial automation is characterized by the technical realization of SOA based on web standards. First approaches for this have been developed in the mentioned research projects. However, the more challenging problem is the organizational aspect of a SOA. Since the standardized communication architecture enables a high degree of flexibility, the complexity of the organization of such an adaptive automated environment also rises significantly. More precisely, the main organizational question and at the same time the most crucial point for a SOA is how a seamless and uncomplicated mapping from the high-level process to the basic services can be realized. Only when the organizational and technical parts work hand-in-hand the true potential of SOA can be leveraged. In the context of production automation this implies a seamless proceeding from the phase of process planning to the execution of the process by the automated manufacturing equipment. Thus, we will present a comprehensive approach that constitutes the fundament for an effective application of SOA for industrial automation by incorporating the process planning.

2. BENEFITS AND SPECIFICS OF SERVICE-ORIENTED AUTOMATION

The use of the paradigm of service-oriented architectures in the context of industrial automation systems is intended to significantly decrease the effort for integration and programming of automation components. The fundamental idea is to achieve a holistic automation architecture enabled by concepts and technologies based on service-oriented architectures. Our approach for this kind of architecture is named SOA-AT what stands for SOA in automation technology.
The two basic aspects for SOA-AT are the standardization of communication interfaces and the functional encapsulation of mechatronic functions that enable a high flexibility and an improved control of complexity for automation systems. In general, every mechatronic function is represented by a software module called service. The functional encapsulation of elementary functions provided by field devices is defined as basic services. Control programs are encapsulated as composed services because they consist of an arrangement of services.
These services are the core elements of the SOA-AT automation architecture. They can be invoked by a service user, called service client. In order that the client knows how to interact with the service, the interface of a service has to be described. If every service interface is described in the same way, for example by using the WSDL standard, a standardization of the communication interfaces can be achieved. As an illustration in Fig. 1, encapsulated functions of field devices "read" could be a service of a RFID reader and "move out" a service of a cylinder. These services can then be invoked within control tasks that are encapsulated again, like the composed service "process 1".

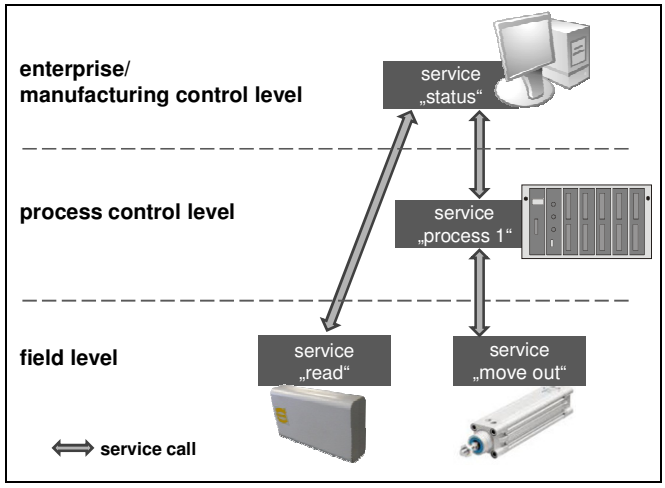

Fig. 1 Service-oriented automation

This example already shows which profits can be achieved with SOA-AT. Through standardized interfaces a level spanning communication of automation components is possible. A client just needs to know the service description of the service it wants to use. The connection of different types of automation components can be done without high integration effort. Thus, the rigid hierarchy of the automation pyramid gets more flexible. Furthermore, the type of programming can be lifted to a higher level. The implementation of a process can be realized through the logical order of service calls. There is no more handling with mere I/O-signals needed because processes are implemented on a functional view. In ideal case the service descriptions of the basic field device services are vendor independent and uniform. Since then the hardware can be chosen and switched very flexibly, a high degree of hardware independence and reusing of devices is possible. This enables a hardware

independent programming style because the device that executes the respective mechatronic function can be assigned later.

Thus, today's automation technology has areas with significant scope for improvement by using paradigms of service-oriented architectures. We want to achieve these potentials by transferring the idea of SOA from its classical field of application to the industrial automation domain. Therefore, we analyzed which properties SOA for enterprise applications has, called SOA-IT, and how these characteristics can be transferred to the industrial automation domain in order to define our approach of SOA-AT.

First, SOA in general is defined as a system architecture that represents versatile, different methods and applications as reusable and openly accessible services that enable a platform and implementation independent usage [Melzer 2008]. Moreover, SOA is neither a technology nor a technology standard. It represents a technology independent, high-level concept that provides architectural blueprints [Krafzig *et al.* 2004].

For SOA-IT these blueprints are focusing on the slicing and composition of enterprise applications as services which are technically independent and have a direct relationship to business functionality. Thus, a service is a software component that executes a special functionality within a business process. In the automation domain a service encapsulates mechatronic functionality. The main difference between these two application fields is the dependence on the location of the hardware that executes the service. While it does not matter where the physical execution of a piece of software takes place, the right execution of a mechatronic function that impact a physical process is heavily dependent on the location of the hardware.

An important characteristic of SOA is the loose coupling of services. The term coupling refers to the degree to which services depend on each other. An agile service environment needs services that are independent to each other. Thus, service compositions can be done quickly and with high flexibility. In normal software architectures loose coupling can mainly be obtained by software modularization and a high degree of cohesion. However, loose coupling of technical processes additionally requires modular design of the mechatronic components in order to rearrange the hardware structure easily. Thus, the hardware of the automation devices and the control programs have to satisfy the paradigm of loose coupling.

Another difference between SOA-IT and SOA-AT exists in the scope of the specification of a service. The service client that wants to use the service needs to know which exact functionality the service implies. This information is normally provided by the service contract. With its help the purpose, constraints, and usage of the service are specified. These kinds of information are sufficient for describing a service that has just software functionality. In the case of a service that encapsulates a mechatronic function of a device we need additional information. As mentioned before, the location of the hardware is essential. So, for the selection of the right service information about the location of its hardware is necessary. Furthermore, if services are reorganized or added so that the hardware structure has to be changed additional information about the hardware is needed. For example, this contains the geometric dimensions or the power supply interfaces of the device.

All in all, we can assert that the basic objectives of SOA-IT and SOA-AT are different. The classical SOA approach divides one big business process in several independent sub processes that are represented as services. The basic services encapsulate software components that execute several parts of a greater data-processed operation. Hence, SOA-IT is a concept for distributed computing. Indeed, a production process has to be implemented through the composition of single basic services, too. However, the outstanding objective is the realization of a real physical process. So, the challenge consists of deriving a well-defined technical process out from an abstract process description.

	SOA-IT	SOA-AT
field of application	business processes	technical processes
service definition	encapsulation of software	encapsulation of mechatronic functionality
location	independent of location of service provider	dependent on location of service provider
loose coupling	modular programming with high cohesion	modular programming + mechatronic design
service specification	software functionality	mechatronic functionality + hardware description + location
objective	distributed computing	execution of a technical process

Fig. 2 Comparison of SOA-IT and SOA-AT

3. PROCESS-ORIENTED FACTORY PLANNING IN COMBINATION WITH SOA-AT

The strategic goal of factory planning processes is to avoid preventable mistakes in the conceptual design of factory systems. Since today's market conditions are continuously changing, the whole manufacturing system has to be frequently reconfigured. Thus, one of the most important targets of factory planning is the assurance of high flexibility and agility of the factory system for minimizing the reconfiguration effort. Moreover, suitable planning and realization methods are needed that enable an efficient reconfiguration process. Today, factory planning processes comprise several development steps. During the detail planning phase, the planning of the mechanical and electrical elements takes place. After that follows the implementation of the control programs. At this point typically a sharp break occurs concerning the information flow. The data of the previous planning processes are not taken over into the planning of the control systems. Consequential, some planning work has to be done twice and the parallel developed results of the different planning domains do not fit together.

This data and information break can be minimized by using a holistic planning method. Therefore, the approach of a process-oriented planning process [Pohlmann 2008] is a suitable planning method. This approach represents a top-down planning procedure that focuses on a functional description of the production process. This functional description bases on the orchestrated functionalities of intelligent field devices or mechatronic modules. While SOA-AT encapsulates the mechatronic functions provided by production equipment into services and guarantees

interoperability it complements the process-oriented planning method. The combination of process-oriented production planning and a SOA-AT based automation architecture enables a unique connection between production planning and the control tasks of the production plant.
In order to enable a process-oriented planning the concept of abstract and concrete services is defined. An abstract service describes the functionality of a service e.g. which operations can be executed and how an interface is formed. The concrete part of a service describes the technical details of communication interfaces, the location a service can be executed, and the name of the service itself. Such services are used both in the planning phase and in the realization phase. They are combined in a process-oriented way in order to set up complete factory systems. The combination of abstract and concrete services allows for bridging the gap between planning, control, and realization. Setting up workflows of abstract services in factory planning phase and combining them with concrete services of mechatronic modules in the realization phase brings the continuous keeping of data. Thereby, a holistic process-oriented planning process can be achieved.

4. THE PROCESS-ORIENTED FACTORY MODEL

In order to realize a process-oriented planning process in combination with a service-oriented automation architecture a suitable models and methods that support the practical implementation and seamless connection of the concepts is necessary. Thus, we developed a process-oriented factory model which constitutes the foundation for planning and control of production plants based on SOA-AT principles. The objective of this model is to support the lifecycle of a production plant from planning and reconstruction to realization and control tasks in a universal and holistic manner. This reference model consists of three parts: the physical model, the operation model, and the task description.
The physical model depicts the structure of the manufacturing equipment. As we illustrated before, SOA-AT is in contrast to SOA-IT characterized by its dependence on the location of the service. Hence, the single services within the production process have to be assigned to the hardware that executes the physical process. Otherwise, if some devices provide the same services it is ambiguous which device has to be invoked. Thus, a mere modeling of the services without a reference to the arrangement of the hardware is not sufficient. The physical model depicts the manufacturing plant in a number of levels, where every level has a special object type. For example, the level "Working Cell Level" contains the objects "Filling Cell" and "Drilling Cell", as shown in Fig. 3. The level beneath contains the objects the cells comprises, as in the example the "Filling Cell" itself comprises a RFID reader, a sensor, and a cylinder. Since there exist already guidelines and standards which define physical models for production plants, we propose the notation of the standard for Enterprise-control system integration [IEC 62264].

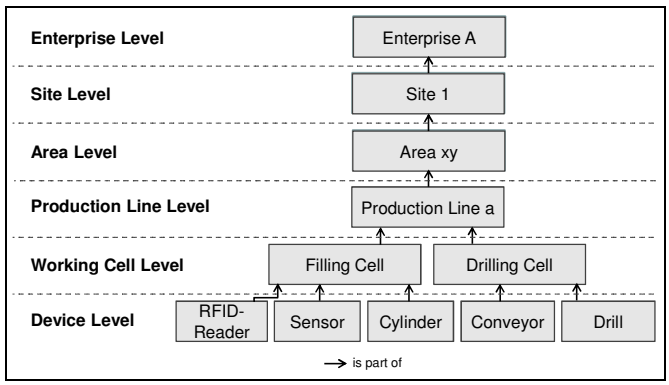

Fig. 3 Example of a physical model of a production plant

The second part, the operation model, depicts the functional structure of the plant. An operation represents a function or a process that is executed by a plant component. So, a high-level operation comprises of other low-level operations again. For that purpose, the operation model is built up in levels similar to the physical model. Hence, a production process can be described by the hierarchically arranged operations that are executed by the manufacturing equipment and control components. Each operation can then be assigned to an object within the physical model in which the operation is executed. Therefore, the operation model does not necessarily have to contain the same levels as the physical model.
However, we suggest creating an obligatory "basic operation level" in which the basic operations of the individual field devices are located. The reason for this is the fact that the operations on the lowest level are the foundation of all operations above. In reality these operations will be executed by the field devices of the plant so that the basic operations represent the elementary functions of field devices. In contradistinction to composed operations the basic operations may not depend on the special process or context in which they will be applied. Actually, in ideal case these operations of field devices are standardized, well-defined and accessible as a service. The overlying levels can be chosen more flexible because their composed operations just comprise basic operations and other composed operations.
An example of an operation that can be linked to the filling cell could be the operation "filling_bin" as shown in Fig. 4. This operation could be split off in two parts "detect_carrier" and "move_objects". These operations still belong to the object "Filling Cell" in the physical model because they are not basic operations. So, we have to assign basic operations that will be realized through field devices. In this case the operation "detect_carrier" could be realized with the basic operation "detect" that will be assign to a sensor in the physical model. The second operation "move_objects" can be realized through a cylinder that moves objects into the bin. The basic operations of the cylinder are "release" and "close". But why do we call these functional units operations and not already services? The operations just model the functional context of a plant for a special process. So, the operations are not yet implemented services that can be truly invoked within the plant. Furthermore, the composed operations do not give any information about the implementation of the process. Thus, we need some additional information about the production task. This information contains the third part of

the overall model, the task model. Within the task model the process of every composed operation is described, as shown on the right hand side of Fig. 4. Therefore, the operations that build a composed operation have to be arranged in the special logical order that characterizes the process. Since there are plenty of ways for modeling processes, we suggest using an existing formal modeling language like sequential function charts or BPMN [OMG 2009].

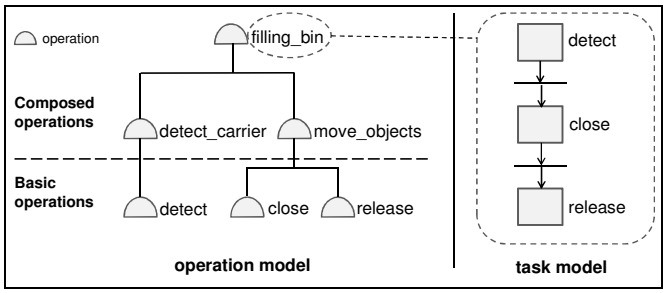

Fig. 4 Example of an operation model in combination with a task description

In which order the models have to be created and linked to each other depends on the planning or reconfiguration task. After the process-oriented model of a production plant has been developed the SOA-AT control architecture can be deduced from the model. Therefore, the basic operations have to be assigned to field devices within the physical model. This combination of an abstract operation with a real device that provides the operation as a service builds the basic field device services. As before said, in ideal case the operations are standardized so that they are conform to the services of the field devices. The composed services are deduced in a similar way. If we want to define a service for a control unit for a special part of the plant, we can choose an operation that is suitable with the help of the physical model. Then, the service orchestration can be extracted from the task description of the operation.

5. EXPERIMENTAL SETUP

Although a lot of research is carried out in the field of service-oriented architectures for production systems, the practical application of service-oriented automation concepts are still far away from copious use in production environments. Hence, existing technologies and concepts have to be evaluated and further developed with the help of experimental setups. Therefore, we work on an experimental setup that is the basis for evaluating SOA technologies and testing our new process-oriented planning concept in connection with service-oriented automation technology.

The experimental setup described below elucidates the far ranging effects of this new architecture and the technologies necessary to implement it. This setup considers the results of the EU project SOCRADES [Karnouskos *et al.* 2009] and other SOA technologies.

The experimental setup in Fig. 5 illustrates an industrial process in which pills can be filled in bins according to the order information stored on a RFID tag which is attached to the bin [Stephan *et al.* 2010]. For a faster comprehension some selected devices are named in Fig. 5. The bin carrier traverses the filling-process form the left to the right. During the next process step the correct amount of pills can be checked by a camera system which compares the detected pills with the order information stored on the RFID-tag. The whole process is accomplished by industrial equipment like motors, frequency converters (ProfiBus-Interface), conveyers, RFID-read-write-units (RS232-Interface), pneumatic stoppers, a pneumatic valve terminal (ProfiBus-Interface), inductive sensors (digital I/O-Interface), ultrasonic sensors (digital I/O-Interface), camera system (Ethernet-Interface). Instead of controlling the whole production unit via a single PLC all devices are controlled via services over Ethernet.

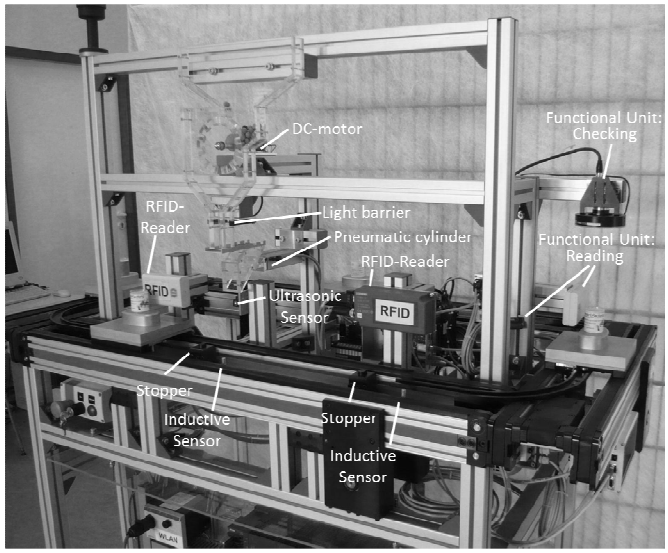

Fig. 5 Experimental setup with SOA-AT field devices

To control the industrial devices via services, gateways are used to translate between the industrial communication protocols and the service protocols. The gateways are realized on different microcontrollers partly with and partly without operating system. To illustrate the service-oriented control approach the following technologies are deployed and compared: UPnP, Web Services (WSDL, SOAP) and DPWS (WS-Addressing, WS-Discovery, WS-Eventing). Universal Plug and Play (UPnP) is an internet protocol based set of network protocols to perform manufacturer-independent control of devices with or without a central control unit. In contrast to Web Services and Device Profile for Web Services (DPWS) it was specially designed to simplify and speed up the use of home entertainment equipment based on the Plug and Play idea from Microsoft Windows. Web Services standardized by the World Wide Web Consortium (W3C) focus on facilitating the cooperation of distributed application programs via the internet [W3C 2004]. DPWS combines and confines a set of W3C standards to allow the deployment of Web Services on embedded systems with restricted hardware resources [OASIS 2009].

Fig. 6 illustrates the physical hierarchy of the experimental setup in combination with its services. The industrial process of filling and checking pills is executed by field devices which are controlled by three gateways and one PC, which is a deputy for a PLC providing a web service interface. The whole process is controlled by the orchestration of the single services via a BPEL engine (Business Process Execution Language) running on the PC.

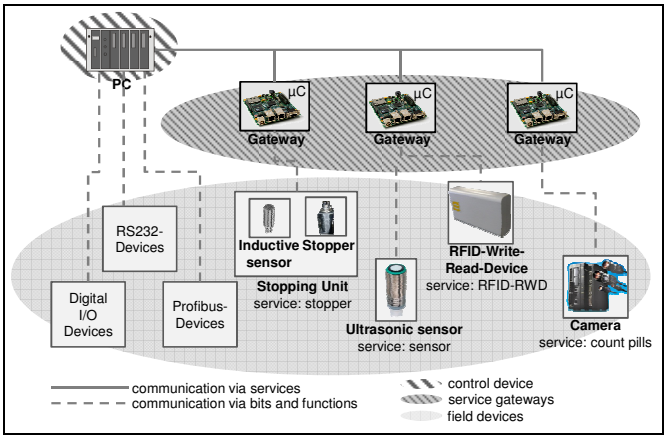

Fig. 6 Physical hierarchy of the experimental setup in combination with its services

This experimental setup reveals the applicability of web-standards for production systems and the feasibility of our process-oriented factory model. The setup is our fundament for evaluating the process-oriented planning concept and the application of our factory model. Further work will deal with the integration of the planning data-sets into the factory model without any media interruption to support the orchestration of industrial production processes. The continuous use of planning data-sets, e.g. of mechanics, electrics, and process plans and their use for process control is the key to improve the agility of production systems.

6. CONCLUSIONS

In this paper we are showing that service-oriented control architectures support modularization and establish interoperability between several automation systems. The architectural paradigm continues the recent developments in automation technology concerning decentralization and growing computation power of field components. Based on process-oriented planning concepts in combination with SOA-AT quick integration and reuse of production equipment can be enabled. Hence, implementing service-oriented automation systems can increase agility of production systems and minimize reconfiguration costs.

However, until the paradigm will be used on a broad basis in today's companies there is still a long way to go. Besides the technical availability of field devices equipped with adequate service-oriented interfaces, which still isn't the case, today's planning processes will have to be adapted to the new possibilities. Up to now, in order to implement a production process software and hardware are linked together in the early development processes. From this point on the planning process is focused on hardware instead on processes. The crucial factor for a successful application of a process-oriented planning method combined with service-oriented control systems is to consequently keep the process-oriented view throughout from the early planning stages to the initial operation of the production line.

Another important factor which can hinder the adoption of this new architectural paradigm concerns the human dimension. For decades factory planners and technicians have gained lots of experiences with conventional PLC and field bus technology for automation purposes. So, technicians and factory planners might not be looking for new paradigms but will see the higher level of abstraction as additional source for project risks. However, we believe that the advantages will drive service-oriented automation architectures from the level of ERP Systems down to MES and last but not least to device level.

7. REFERENCES

Karnouskos, S.; Bangemann, T. and Diedrich, C. (2009). *Integration of Legacy Devices in the Future.* INCOM09 the 13th IFAC Symposium on Information Control Problems in Manufacturing, Moscow, Russia

Krafzig, D., Banke, K., and Slama, D. (2004). *Enterprise SOA: Service-Oriented Architecture Best Practices.* Prentice Hall, Upper Saddle River, New Jersey, USA.

Melzer, I. (2008). *Service-orientierte Architekturen mit Web Services: Konzepte – Standards – Praxis.* Spektrum Akademischer Verlag, Heidelberg, Germany.

OASIS Web Services Discovery and Web Services Devices Profile (WS-DD) TC (2009). *OASIS Devices Profile for Web Services (DPWS) Version 1.1*. Oasis Standard. http://docs.oasis-open.org/ws-dd/dpws/1.1/os/wsdd-dpws-1.1-spec-os.pdf.

Object Management Group (OMG) (2009). *Business Process Model and Notation (BPMN)*. OMG Document Number formal/2009-01-03, http://www.omg.org/spec/BPMN/1.2.

PABADIS'PROMISE Consortium (2008). *Structure and Behaviour of a PABADIS'PROMISE System.* White Paper. http://www.uni-magdeburg.de/iaf/cvs/pabadispromise/dokumente/whitepaper2_v60.pdf.

Pohlmann, E. (2008). *Methodik zur prozessorientierten Planung serviceorientierter Fabriksteuerungssysteme.* Fortschritt-Berichte pak 17, Universität Kaiserslautern, Kaiserslautern, Germany.

Souza, L.; et. al. (2008): *A Web Service based Shop Floor Infrastructure.* Internet of Things 2008, Conference proceedings pp. 50-67, Zurich, Switzerland.

Stephan, P. et. al. (2010). *Product-Mediated Communication through Digital Object Memories in Heterogeneous Value Chains*. IEEE International Conference on Pervasive Computing and Communications (PerCom). Conference proceedings 199-207, Mannheim, Germany.

Vogel-Heuser, B. et al. (2009). *Global Information Architecture for Industrial Automation*. In: atp 1-2.2009. On pages 108 – 115, Oldenbourg Verlag, München, Germany.

W3C Working group (2004). *Web Services Architecture*. Working Group Note, http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/.