Subject-Oriented BPM as the Glue for Integrating Enterprise Processes in Smart Factories

Udo Kannengiesser[1], Matthias Neubauer[2], and Richard Heininger[1(✉)]

[1] Metasonic GmbH, Münchner Str. 29 - Hettenshausen 85276, Pfaffenhofen, Germany
{udo.kannengiesser,richard.heininger}@metasonic.de
[2] Johannes Kepler Universität Linz, Institut für Wirtschaftsinformatik – Communications Engineering, Altenbergerstaße 69 4040, Linz, Austria
matthias.neubauer@jku.at

Abstract. This paper presents how an existing approach to business process management, Subject-oriented BPM (S-BPM), provides a foundation for seamlessly integrating processes in production enterprises, from business processes to real-time production processes. The applicability of S-BPM is based on its simplicity and encapsulation of separate process domains. This supports agility as all stakeholders can be engaged and the effects of changes can be limited to individual modules of the process. An application and tool support developed in an ongoing European research project are presented to illustrate the approach.

Keywords: Seamless process integration · Smart factories · S-BPM · OPC UA

1 Introduction

As industry moves towards cyber-physical production systems (CPPS) and internet of things (IoT) manufacturing, the ways in which enterprise processes are conceptualised and executed are changing. Decentralising production using large numbers of interlinked cyber-physical production resources breaks up the traditional boundaries drawn between different process abstraction layers. Smart devices and processes at all levels in the industrial control hierarchy need to interact if the vision of vertically and horizontally integrated production systems is to be realised [1, 2]. A number of integration approaches have been defined for this purpose, including IEC 62541 [3] and IEC 62264 [4].

Despite the availability and wide acceptance of such standards, designing integrated enterprise processes involving cyber-physical systems remains a challenge [5]. One issue is the lack of a process modelling language that is both understandable by all stakeholders and formally defined to allow model-driven execution. Most work on process modelling for production enterprises draws on existing approaches from the domains of software engineering and business process management (BPM). Examples include UML [6], BPMN [7, 8] and Petri Nets [9]. Yet, most of the current BPM approaches are not sufficiently formal to be immediately executed [10] and are difficult to be learned and applied by untrained modellers [11] such as business people and

I. Ciuciu et al. (Eds.): OTM 2015 Workshops, LNCS 9416, pp. 77–86, 2015.
DOI: 10.1007/978-3-319-26138-6_11

shopfloor workers. In addition, the centralised control-flow paradigm underpinning these modelling approaches is at odds with the decentralised control advocated for smart factories [5, 12]. Small changes in the production process (e.g. when a new product variant needs to be manufactured) can be quite difficult to implement and verify in a monolithic (centralised) process model. As agility is a principal motivation for smart factories [12, 13], there is a need for alternative approaches to integrated process modelling.

This paper shows that subject-oriented BPM (S-BPM) [14] provides such an approach. S-BPM has two distinguishing features that support agile processes:

1. Simple yet formal notation: S-BPM provides a common (and executable) language for all stakeholders including process participants not trained in a specialised modelling notation.
2. Encapsulation: S-BPM supports modular process architectures as it separates the concerns of different process domains and encapsulates them in "subjects" that are loosely coupled via messages.

Section 2 introduces S-BPM and highlights these features using examples from the business process domain. Section 3 explains how S-BPM can be applied for integrating business and production processes. Section 4 illustrates this approach based on prototypes developed within the Factories of the Future project SO-PC-Pro. Section 5 concludes the paper.

2 Subject-Oriented Business Process Management

Subject-oriented Business Process Management (S-BPM) [14] was first proposed by Albert Fleischmann in 1994 [15]. Formally, the S-BPM approach is based on the parallel activity specification scheme (PASS) which extends elements of the Calculus of Communicating Systems by Milner and Communicating Sequential Processes by Hoare [14, p.289f], This approach differs from traditional process modelling methods in that it is based on a decentralised view: Processes are understood as interactions between process-centric roles (called "subjects"), where every subject encapsulates its own behaviour specification [11]. Subjects coordinate their individual behaviours by exchanging messages. Such a communication-based approach differs from traditional BPM paradigms that require the orchestration of activities via tokens being passed along a central control flow. Messages in S-BPM may include information at any level of granularity, from simple notifications or requests to complex data structures (referred to as business objects).

2.1 Notation

S-BPM models include two types of diagrams: A Subject Interaction Diagram (SID) specifying a set of subjects and the messages exchanged between them, and a Subject Behaviour Diagram (SBD) for every subject specifying the details of its behaviour. SBDs describe subject behaviour using state machines, in which every state represents an action. There are three types of states in S-BPM: "receive" states for receiving messages, "send" states for sending messages, and "function" states for performing actions operating on business objects (i.e., actions performed without involving other subjects). Examples of a SID and a SBD are shown in Fig. 1 and Fig. 2, respectively. Details of the notational elements used can be found in [14].

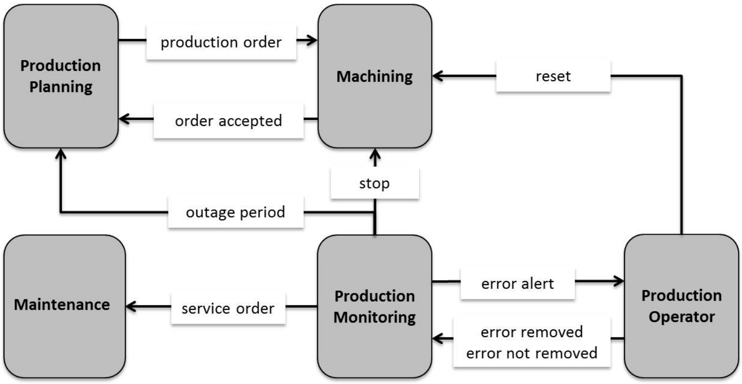

Fig. 1. Subject Interaction Diagram (SID) showing the communication between subjects in a production context

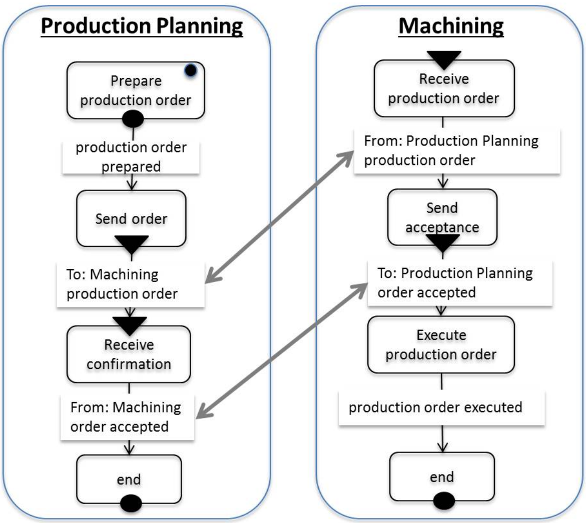

Fig. 2. Subject Behaviour Diagrams (SBD) showing the individual behaviours of the "Production Planning" subject and the "Machining" subject. Their interconnections through pairs of "send" and "receive" actions are represented using double-headed arrows (not part of the S-BPM notation).

Subjects may be executed by human or computational actors [11, 13], including cyber-physical systems or machines as indicated in Fig. 1 and Fig. 2. Their well-defined semantics (based on Abstract State Machines [16]) allow for automatic translation into executable code, including PLC code represented using IEC 61131-3 [17].

The simple notation using only few building blocks allows domain experts without formal training in process modelling to readily create valid S-BPM models. This is a departure from traditional approaches such as BPMN that require either extensive training of all stakeholders or facilitation by an experienced modeller – both of which are time-consuming and costly.

2.2 Encapsulation

Another key characteristic of subject-oriented process models is encapsulation: Every subject in a process model encapsulates a behaviour specification for performing the particular functionality represented by that subject. This idea is reflected in the two types of diagrams in S-BPM. The SID shows only the black-box behaviours of a subject, representing only the messages received and sent. This is similar to the notion of a service in service-oriented architectures (SOA). The detailed behaviour of the subject is usually opaque; for interacting with that subject one only needs to know its functionality (denoted by the subject name) and the messages it can receive and send [18]. The internal behaviour of a subject as described in its SBD is usually visible and modifiable only by the owner of that subject.

The benefit of subjects encapsulating their behaviours is that process models become modular. Changes in the process can be realised more rapidly than using a monolithic process model, as individual modules (i.e. subjects) can be modified without necessarily affecting other subjects. In addition, the responsibilities for providing, monitoring and adapting the functionality of different subjects can be clearly assigned, usually to the respective subject owners.

To illustrate the effect of modularity in subject-oriented process models, consider the example shown in Fig. 3. Here a “Vacation Requestor” subject (owned/executed by

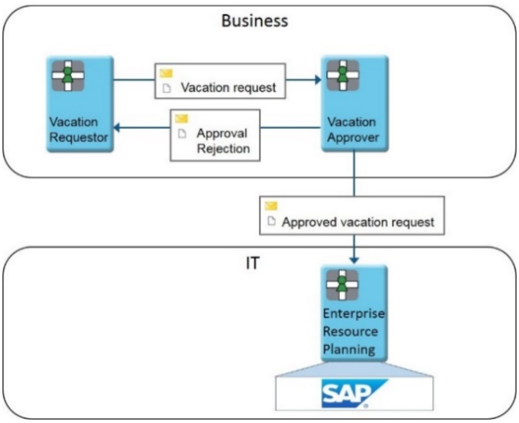

Fig. 3. A vacation process in which business and IT domains are integrated via encapsulation of subject behaviours

a company employee) sends a vacation request to a “Vacation Approver” subject (owned/executed by the employee’s manager) who sends back either an approval or a rejection message. In case of approval, the “Vacation Approver” subject additionally notifies the “Enterprise Resource Planning” subject. That subject is owned by an IT expert, and its behaviour is executed by an SAP system.

If there are changes in the way the approved vacation request needs to be managed by the “Enterprise Resource Planning” subject, the IT expert can modify the internal behaviour of that subject without the other subjects (and their subject owners) having to know or care about – at least as long as the type of message (or the specific business object) conveyed to the subject is not affected by the change.

3 Vertical Process Integration Based on S-BPM

The concept of encapsulation can be applied to the vertical integration of enterprise processes across the different levels of the automation pyramid. The IEC 62264 control hierarchy [4] is shown on the left-hand side of Fig. 4. This hierarchy represents the processes in production companies at four levels: field instrumentation control (Level 1), process control (Level 2), manufacturing operations management (Level 3), and business planning & logistics (Level 4). As these levels impose distinct requirements on processes with respect to real-time processing, data storage, safety and security, the development of models and systems at each level has been undertaken rather independently. This has resulted in poorly integrated applications especially between the Low Level Control domain (LLC, i.e. Levels 1 and 2) operating in real time and the High Level Control domain (HLC, i.e. Levels 3 and 4) operating in non-real time. Systems developed for LLC include Programmable Logic Controllers (PLCs), and systems for HLC include ERP, MES and BPM systems.

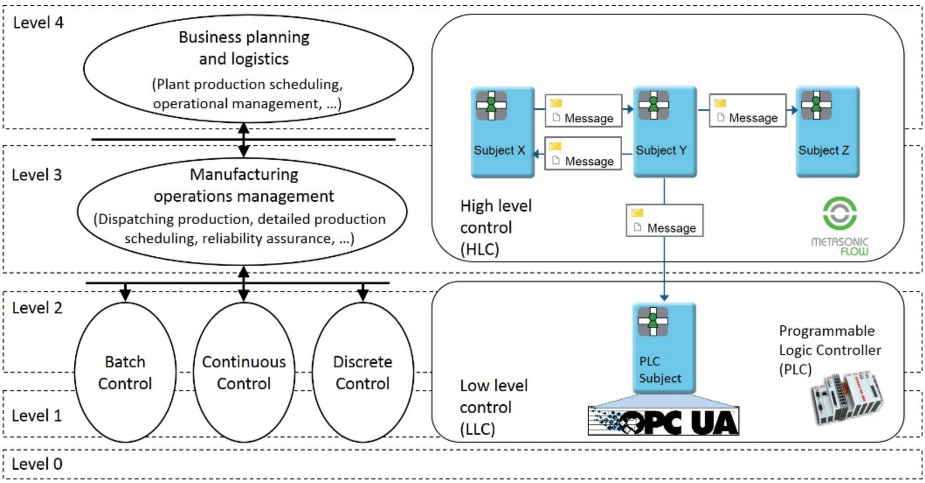

Fig. 4. Seamless process integration across the IEC 62262 control hierarchy (image adapted from [1]), based on subjects encapsulating domain-specific behaviours

A generic S-BPM process model is shown on the right-hand side of Fig. 4, encapsulating a LLC behaviour in a “PLC” subject. That subject is owned by an automation engineer, and its behaviour is executed by a workflow engine that communicates with a PLC via the OPC UA (IEC 62541) [3] standard.

When changes in LLC processes become necessary (e.g. when the control software of a production machine needs to be reconfigured), they can be easily managed by modifying the internal behaviour of the PLC subject and checking whether messages with other subjects need to be adapted. If messages need to be changed, the respective subject owners (e.g. a business person and an automation engineer) must come to an agreement on the specific message adaptations. However, they do not need to know about the detailed internal behaviour of each other’s subjects.

4 Prototype

An interface to the OPC UA standard has been developed for the S-BPM tool Metasonic Suite (www.metasonic.de/en) within the EU FP7 funded project SO-PC-Pro (www.so-pc-pro.eu). The basic features of the interface have been derived from the structure of the OPC UA standard [3]. OPC UA applies the fundamental client-server concept to implement the interaction between different communication partners, e.g. a workflow engine and a plant floor PLC. To allow requesting services provided by an OPC UA server or within a network of OPC UA servers, OPC UA defines an AddressSpace model. In such an AddressSpace an OPC UA server defines which contents (i.e. nodes representing objects, variables, methods etc. for real objects) are visible/editable for clients. Servers also allow clients to monitor attributes and events at the server. Every client can subscribe to the attributes and events it is interested in and will then be notified accordingly.

Fig. 5 illustrates the basic features of the prototype using a schematic representation for the interplay between the behaviour of a “PLC subject” in the Metasonic Suite and a PLC addressable via an OPC UA server. Using the prototype, one basically may (1) configure the endpoint of the server, (2) configure the relevant node (e.g. variable, method, and event), (3) read/write variables from/to business objects, (4) invoke methods on the server, and (5) subscribe to data changes or events provided by the server.

The application of the prototype can be illustrated using a simple LED-light switching example. Here the LED lights ‘green’, ‘yellow’, and ‘red’ of a concrete PLC are configured as accessible Boolean variables on an OPC UA server. Furthermore, an S-BPM process for switching the LED lights is defined as shown in the SID in Fig. 6. This process comprises two subjects: The “Light Management” subject specifies the actions relevant for human users (which may be interpreted as a HLC process), and the “Light Controller” subject specifies the behaviour for switching on/off the lights and querying the current light status from the OPC UA server (which may be interpreted as a LLC process).

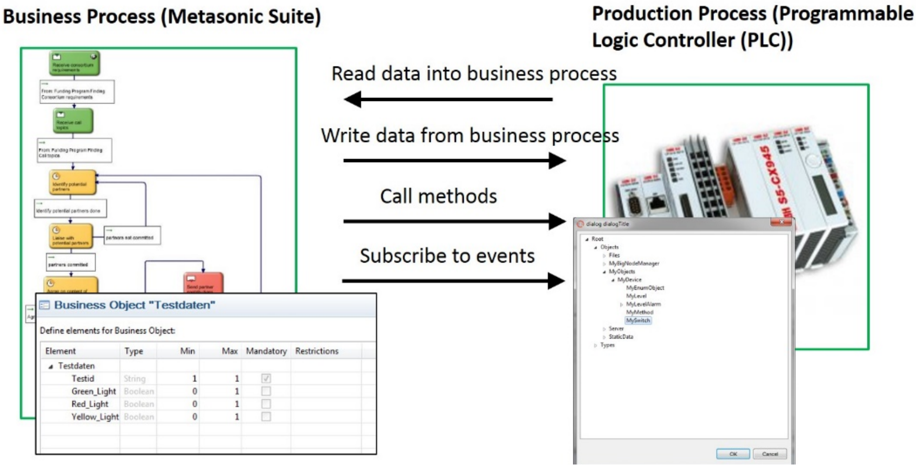

Fig. 5. Schematic feature representation of the OPC UA interface

Fig. 6. Subject Interaction Diagram (SID) of a light switching process

The configuration for accessing the OPC UA services is done by using a so-called "refinement template" in Metasonic Suite: a GUI for configuring data interfaces to external systems. The concrete OPC UA refinement template shown in Fig. 7 allows (i) reading values from a PLC and store them in a business object and (ii) writing concrete values of a business object to variables of a PLC. The template thus facilitates configuring the concrete OPC UA server endpoint that provides the desired variables. Furthermore, one needs to choose the action and the relevant business object before mapping variables to each other. The user interface shown in Fig. 7 allows mapping multiple PLC variables to different fields of business objects.

After modelling and configuring all subject behaviours, the process can be executed in Metasonic's workflow engine. A screenshot of the generic user interface for executing the LED light switching process is shown in Fig. 8. In the example a user may choose one of three options: (1) set lights, (2) request light status, and (3) quit. When clicking "Set lights" the user will proceed to the step "Turn lights on/off" in which the status of the LED lights can be set as desired (i.e., on or off). The desired status will then be sent to the "Light Control" subject that writes this status as a value to the configured OPC UA server at runtime.

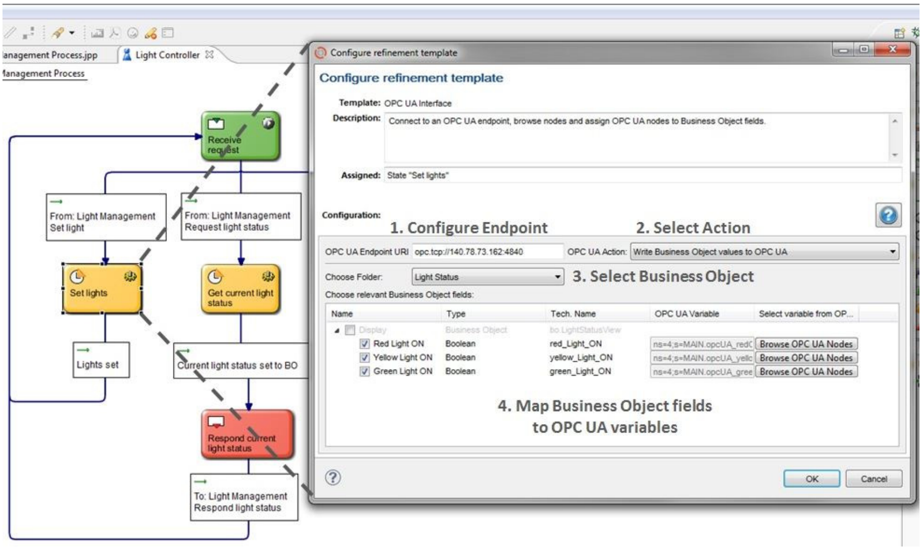

Fig. 7. Refinement template for reading/writing values from/to an OPC UA server variable

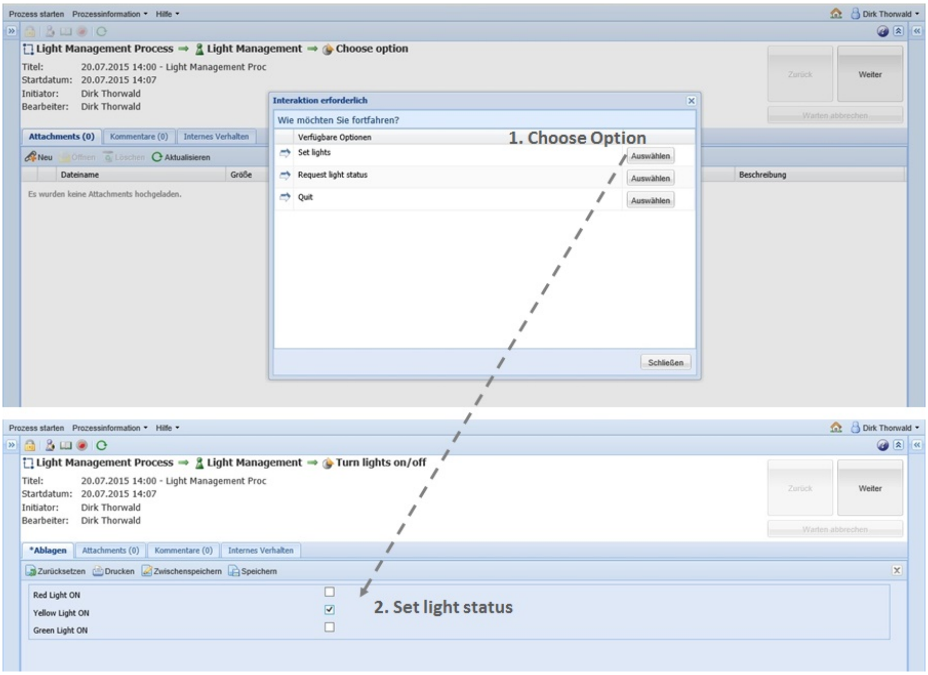

Fig. 8. Executing the light switching process (from the user's perspective)

The presented OPC UA interface is at the prototype stage. It has been tested within four different application scenarios. In the first application scenario a process for managing sun-blinds in a smart home has been modelled and executed. In the second scenario the prototype has been used to manage room lights in offices at different locations. In the third scenario, the power consumption of production machines in a medium-sized manufacturing company has been measured and analysed for process control and improvement. In the fourth scenario, an assembly process has been modelled in which the stress level of a worker (measured using a wearable sensor) is indicated by LED lights (green | yellow | red). Ongoing work comprises the conduction of user tests to improve the user interface and get feedback from process modellers in the field business information systems.

5 Conclusion

In his seminal paper on smart factories Zuehlke [12] concludes with a list of recommendations for future research and development:

- "reduce complexity by strict modularization and lean technologies,
- avoid centralized hierarchies in favour of loosely linked decentralized structures consisting of self-adapting modules,
- allow for self-organization on the system level wherever possible, […]
- create and apply standards to all levels of the automation pyramid in order to reduce planning effort and allow re-use of components,
- and in the end: develop technologies for the human. A deserted factory is an aberration!"

The approach described in this paper follows these recommendations as it uses the modular, decentralised and simple modelling concepts provided by S-BPM, and interfaces with the OPC UA standard. Applying the approach across the entire enterprise can thus lead to process integration that is consistent with the foundational ideas of smart factories. This is a major condition for supporting agile processes in these factories and realising associated business models.

Acknowledgements. The research leading to these results has received funding from the EU Seventh Framework Programme FP7-2013-NMP-ICT-FOF(RTD) under grant agreement n° 609190 (www.so-pc-pro.eu).

References

1. Gerber, T., Theorin, A., Johnsson, C.: Towards a seamless integration between process modeling descriptions at business and production levels: work in progress. Journal of Intelligent Manufacturing **25**(5), 1089–1099 (2014)
2. Colombo, A.W., Karnouskos, S., Bangemann, T.: A system of systems view on collaborative industrial automation. In: 2013 IEEE International Conference on Industrial Technology (ICIT), Cape Town, South Africa, pp. 1968–1975 (2013)

3. IEC 62541-1:2008 OPC Unified Architecture – Part 1: Overview and Concepts. International Electrotechnical Commission, Geneva
4. IEC 62264-1:2013 Enterprise-control system integration – Part 1: Models and terminology. International Electrotechnical Commission, Geneva
5. Horváth, I., Gerritsen, B.H.M.: Outlining nine major design challenges of open, decentralized, adaptive cyber-physical systems. In: Proceedings of the ASME 2013 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Portland, OR (2013)
6. Kohler, H.J., Nickel, U., Niere, J., Zundorf, A.: Integrating UML Diagrams for production control systems. In: International Conference on Software Engineering, pp. 241–251. ACM, New York (2000)
7. Garcia-Dominguez, A., Marcos, M., Medina, I.: A comparison of BPMN 2.0 with other notations for manufacturing processes. Key Engineering Materials **502**, 1–6 (2012)
8. Witsch, M., Vogel-Heuser, B.: Towards a formal specification framework for manufacturing execution systems. IEEE Transactions on Industrial Informatics **8**, 311–320 (2012)
9. Gradisar, D., Music, G.: Production-process modelling based on production-management data: A Petri-Net approach. International Journal of Computer Integrated Manufacturing **20**, 794–810 (2007)
10. Börger, E.: Approaches to modeling business processes: a critical analysis of BPMN, workflow patterns and YAWL. Software & Systems Modeling **11**(3), 305–318 (2012)
11. Fleischmann, A., Kannengiesser, U., Schmidt, W., Stary, C.: Subject-oriented modeling and execution of multi-agent business processes. In: 2013 IEEE/WIC/ACM International Conferences on Web Intelligence (WI) and Intelligent Agent Technology (IAT), Atlanta, GA, pp. 138–145 (2013)
12. Zuehlke, D.: SmartFactory—Towards a factory-of-things. Annual Reviews in Control **34**(1), 129–138 (2010)
13. Kannengiesser, U., Müller, H.: Towards agent-based smart factories. In: 2013 IEEE/WIC/ACM International Conferences on Web Intelligence (WI) and Intelligent Agent Technology (IAT), Atlanta, GA, pp. 83–86 (2013)
14. Fleischmann, A., Schmidt, W., Stary, C., Obermeier, S., Börger, E.: Subject-Oriented Business Process Management. Springer, Berlin (2012)
15. Fleischmann, A.: Distributed Systems – Software Design & Implementation. Springer, Berlin (1994)
16. Börger, E., Stärk, R.: Abstract State Machines: A Method for High-Level System Design and Analysis. Springer, Berlin (2003)
17. Müller, H.: Using S-BPM for PLC code generation and extension of subject-oriented methodology to all layers of modern control systems. In: Stary, C. (ed.) S-BPM ONE 2012. LNBIP, vol. 104, pp. 182–204. Springer, Heidelberg (2012)
18. Kannengiesser, U., Radmayr, M., Heininger, R., Meyer, N.: Generating subject-oriented process models from ad-hoc interactions of cognitive agents. In: 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technology (IAT), Warsaw, Poland, pp. 440–446 (2014)