Industry 4.0 and Object-Oriented Development: Incremental and Architectural Change

Martin Prause *[1], *Juergen Weigand* [1]

Abstract: Industry 4.0 in manufacturing is about combining cyber-physical systems with industrial automation systems. This integration of systems so different in nature aims to create context-aware factories in which people and machines are in real-time alignment. This paper examines the change processes triggered by Industry 4.0 from a conceptual perspective. We find that the observed patterns of change are not novel but have a lot in common with the paradigmatic shift in software development from structured to object-oriented development. The latter approach features to be incremental in the production phase and architectural in the product and process design phase. We argue that Industry 4.0 will be equally paradigmatic and mind-set changing for architects and engineers as to crafting production processes and creating products for the future.

Keywords: Advanced Manufacturing; Industry 4.0; Object-oriented Development; Technological Change; Technological Innovation

Submitted: January 16th 2016 / Approved: Approved: June 7th 2016

Introduction

The future of manufacturing is being shaped by megatrends such as changing demographics, globalization, scarcity of resources, climate change, dynamic technologies and innovations, and mass customization (Abele and Reinhart, 2011). Industry 4.0 is Germany's policy answer (BMBF, 2013) to increasing complexities of manufacturing systems and mounting external environmental challenges (Spath et al., 2013). Proclaimed as the fourth industrial revolution, Industry 4.0 is set to be paradigm and strategy – a novel approach to thinking manufacturing and the way of "how-to" transition from traditionally centralized control structures to decentralized ones (BMBF, 2013). In essence, Industry 4.0 is the intelligent real-time, horizontal and vertical integration of humans and machines with objects and information and communication technology systems ("digitalization") to enable a flexible and dynamic management of complex systems (Bauer et al., 2014). More specifically, Industry 4.0 can be defined as the "[…] integration of cyber-physical-systems in production and logistics as well as the application of Internet of Things in industrial processes. This includes the consequences for the value chain, business models, services and work environment." (Kagermann and Wahlster, 2013).

Digitalizing the production chain is not a new trend. During the early 1990s computer integrated manufacturing (CIM) turned into a hype and gained significant momentum. CIM rested on the idea of a fully automated production process from procurement and production to distribution without necessitating human interaction (Bauernhansl et al., 2014). Its successful implementation however was hampered mainly due to missing information and communication technology (ICT) standards (Spath et al., 2013), insufficient understanding, human resistance to change, organizational incompatibilities, and lack of skilled labour to implement and use CIM (McGaughey and Snyder, 1994).

Today the situation is different. On the social dimension, industries are experiencing a shift from an economy centered on organizations to one centered on the individual (Bermann et al., 2013). Based on assistive (ambient) systems and sophisticated human-to-machine interfaces production process flexibility and employees play a key role in Industry 4.0 (Spath et al., 2013). On the technology front, wireless communication and the Internet of Things have reached industrial maturity (Evans and Annunziata, 2012). Politically, the German federal government pushes the importance of standardization (Pichler and Reinhold, 2015; Wahlster, 2014).

The trend of defining and pursuing advanced manufacturing strategies for national economic growth gained momentum in large economies recently. The President's Council of Advisors on Science and Technology in the United States advised the Government to implement an advanced manufacturing strategy (PCAST, 2011), which was accompanied by a national strategy plan one year later (NCST, 2012). Europe's *Horizon 2020* strategy constitutes a roadmap for the factories of the future based on intelligent manufacturing machines (European Commission, 2013). China's State Council has announced its *Made in China 2025* strategy to upgrade its industrial sector across a broad range of industries focusing on intelligent manufacturing product and process innovations, advanced materials, and green manufacturing (State Council, 2015). According to Dezhina et al. (2015), the Russian government has been prioritizing advanced manufacturing since 2013.

However, this momentum will not radically change manufacturing by tomorrow. According to the President of the German National Academy of Science and Engineering, the impact of Industry 4.0 will be revolutionary but its diffusion 4.0 is likely to be evolutionary (Spath et al., 2013).

(1) WHU – Otto Beisheim School of Management Burgplatz 2, Campus Vallendar, Germany
*Corresponding author: martin.prause@whu.edu

A similar paradigmatic revolutionary change with evolutionary transition was experienced by the software industry which transitioned between the early 1970s and the 1990s from structured system development to object oriented system development. In structured system development, the overall process is factored into modules, and each module in the system represents a step in the overall process. The overall process and its decomposition into modules can be represented as a structured top-down chart. In the object-oriented approach, the structure of the system is created around the objects that exist in the model of reality (problem domain). The overall process can be represented as a set of interacting objects (Booch, 1986). Objects represent elements of the problem domain. Different concepts and tools have been developed to support this paradigm in the design phase (object oriented design, OOD) and in the implementation phase (object-oriented programming, OOP).

Similar determined efforts have been made recently in the area of industrial automation systems (IAS). With the specification of a digital factory by the International Electrotechnical Commission (IEC 62832) and the upgrade of the standard specification for programmable controls (IEC 61131 – Version 3), object oriented paradigms were introduced to industrial automation systems in design and implementation phases. Preliminary reference models for Industry 4.0 are directly created using an object oriented structure (Adolphs et al., 2015).

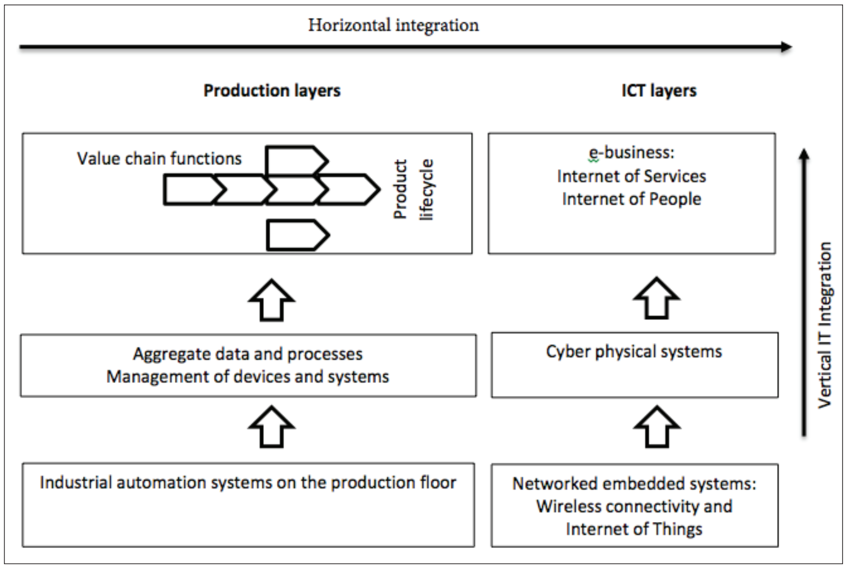
The present paper compares the Industry 4.0 approach and object-oriented development, conceptualizes the resulting system changes, and derives management implications.

Core principles of Industry 4.0

Today's industrial automation systems comprise of a plant to perform the physical process (physical world). Connected and embedded computers in combination with software systems (cyber or virtual world) control and monitor the production process by receiving and analysing inputs from the plant and regulating the site by the computational results (Thramboulidis, 2015). Due to the increase of external complexity originating from megatrends and internal complexity such as increasing product-, customer- and supplier-portfolios, new materials, production processes and IT systems, manufacturing companies have to balance inner and outer complexity to remain competitive (Bauer et al., 2014). The balancing act of increasing product variety and decreasing production batches increases the challenges for a centrally controlled production system. Decentralization facilitates the reduction of complexity (Spath et al., 2013).

The so-called Smart Factory is a specific deployment of the Industry 4.0. The Smart Factory is modularized, self-regulatory (self-adaptive) and digitally integrated with all business functions, within and beyond the organizational boundaries. A Smart factory comprises of intelligent sensor and actor systems (cyber-physical system) to facilitate context sensitive production processes and ICT-based integration of this system across the value chain, value network, and product lifecycle. The Smart Factory is based on transferring the idea of ubiquitous (wireless) computing (Weiser, 1991) to an industrial context (Zuehlke, 2010). A Smart Factory is a "[…] Factory that context aware assists people and machines in execution of their tasks […] by systems working in background, so-called calm systems and context-aware applications" (Lucke et al., 2008). A cyber-physical system (CPS) includes sensors and actors to recognize objects, machines and humans in the production environment to trigger actions based on the environmental context. Successful integration of CPS in the context of Industry 4.0 requires firms to focus on "[…] horizontal integration through value networks, end-to-end digital integration of engineering across the entire value chain and vertical integration and networked manufacturing systems" (Kagermann and Wahlster, 2013) (cf. Figure 1).

Figure 1: The basic concept of Industry 4.0: Connecting machines, products and humans along the horizontal production layers and vertical ICT layers


Horizontal integration
Production layers
ICT layers
Value chain functions
Product lifecycle
e-business: Internet of Services Internet of People
Aggregate data and processes Management of devices and systems
Cyber physical systems
Vertical IT Integration
Industrial automation systems on the production floor
Networked embedded systems: Wireless connectivity and Internet of Things

Cyber-physical systems vertically integrate the physical world of production plants and embedded devices, based on the Internet of Things, with the virtual world of business processes, the Internet of Services, and social network systems for human-machine communication, the Internet of People. In this concept, objects, machines, and humans have two abilities: (1) an inner state and (2) the capability to communicate the inner state. It implies that products can communicate with their environment (at least passively) as so-called *Smart Products*. A Smart Product is not only uniquely identifiable, but it also has additional attributes to reflect its state and a plan of action for the next production step based on its current context (Vogel-Heuser, 2014; Wahlster, 2014). Depending on its context, which is monitored by sensors and actors, the smart product provides the machine with the logic to process the product. Put differently, the logic of the production process is partly decomposed to the interacting elements. This concept induces the changes from a structured centralized (top-down) control to a decentralized (bottom-up) control, based on the state of and interaction between products, machines, and humans.

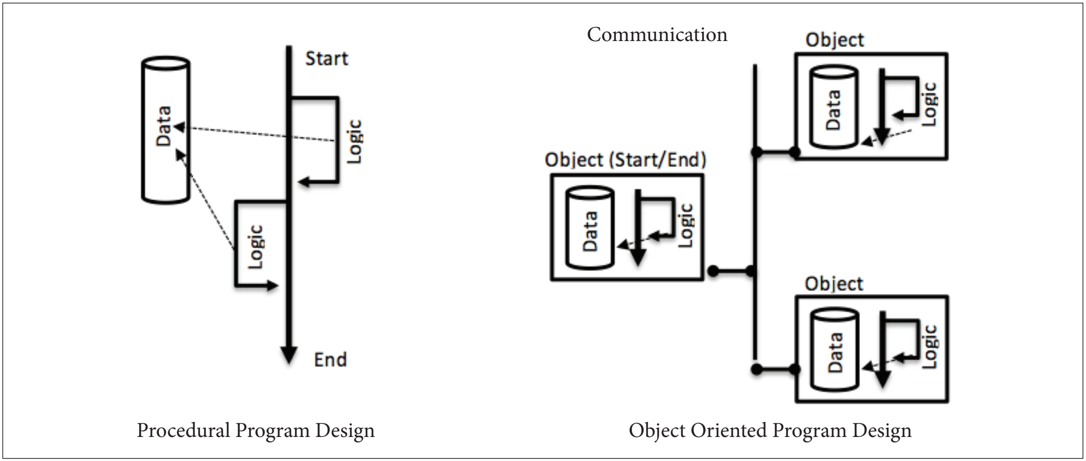
Object-oriented development

Object-oriented programming was first drafted by Ole-Johan Dahl and Kristen Nygaard in 1961 by designing the programming language Simula (Dahl, 1981). The idea evolved and became finally prominent in 1972 with the development of the programming language Smalltalk by Xerox (Capretz, 2003). Since then and for the ensuing 25 years, object-oriented programming took an evolutionary path before peaking in the 1990th (Sircar et al., 2001). Until object-oriented programming became popular, the software was written in a procedural way. Its logic and data was centrally structured. The logic defines how the data must be modified. Data and logic were kept separately. With the exponential increase of computational power two challenges emerged. First, the execution of code became more efficient. Therefore, programs of higher complexity could be created. Consequently, it became increasingly difficult to keep track of and maintain the entire program logic, which led to a software crisis in the early 1990s. The relative cost of maintenance and development had by far exceeded hardware costs (Selvi et al., 2009). Second, to speed up computing parallel execution of code was desired but hard to implement, because the centralized logic and data structure could hardly be split up to be served by different machines.

Object-oriented development proved a solution to these complexity and concurrency challenges. Elements of a problem (model of reality) are represented by objects that mimic elements of the problem domain and just solve sub-problems (Bitter and Rick et al, 2001). An object comprises of internal data to reflect its state and a logic to modify the data to solve the sub-problem. Thus data and logic are highly cohesive. Both data and logic are encapsulated in a *class*. A class is a general definition or template of an element of the problem domain. During the execution of a program, objects are created from a class. These objects are specific instances of a class. For example, assume a class *Dog* with specific attributes e.g. *can bark* and *has four legs*. While the class *Dog* is a general definition of a dog, an instantiated object of the class *Dog* would be a *Terrier*, *Collie* or *Shepard*. All instantiated objects have the specified attributes in common (*can bark* and *has four legs*) and probably more specific attributes. Data and logic are restrictively accessible within the object scope (encapsulation), which leads to low cohesiveness among objects. To communicate with its environment, an object provides a service, so-called *methods*, which can be invoked by other objects to exchange data or trigger some object-specific behavior. A top-level program handles the communication between the loosely cohesive objects and combines the results of the solved sub-problems. As long as the program can handle the class Dog, it can handle any object (*Terrier*, *Collie* or *Shepard*) based on the class dog. The difference between procedural program design and object oriented program design is depicted in Figure 2.

While object-oriented programming has many other features such as inheritance or modularization or polymorphism etc. (Dahl, 1981) to reduce complexity (increase reusability) and enhance concurrency, the key element is based on abstraction and encapsulation.

Figure 2. In the procedural design pattern the logic of the program is structured in procedures (sub-routines) which all access a common data set. In object-oriented design, each object has its own data set. The abstraction is modeled by the boundaries of an object and the communication element. The inner logic of an object, how to solve a problem, is not visible to others.

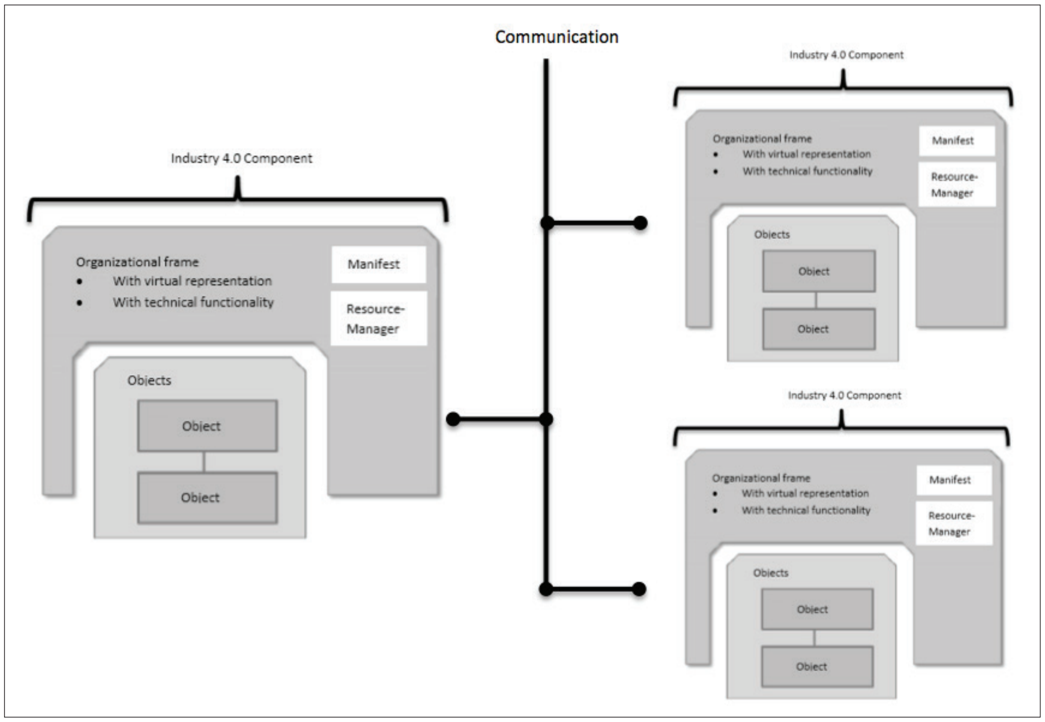


Conceptual similarities between Industry 4.0 and object-oriented development

The change processes in software development and presently in manufacturing have the same root cause. Increasing complexity forces the system to change from centrally-structured control to decentrally-structured control, with a higher degree of autonomy, self-regulation, and self-optimization. This fragmentation of the value chain has been already postulated in the context of fractal factories in 1995 (Warnecke, 1995). Both approaches take a bottom-up approach to implement decentralization and shift the program logic to the decomposed elements.

The model of such a decomposed system in Industry 4.0 is based on a reference architecture with its core element, the Industry 4.0 component (Figure 3). Like an object in object-oriented development may represent any element of the problem domain, the Industry 4.0 component may represent either a production system, a single machine, an assembly line group or a product (Adolphs et al., 2015). It comprises of the physical objects (products or materials) and an organizational frame, which encapsulates the physical objects from their environment and handles the communication. The frame contains a virtual representation of the physical objects, the technical functionalities describing the logic of the production steps, and the overhead data. In analogy with the object-oriented paradigm, data and logic are combined and only restrictively accessible within the scope of the object. Among objects, relationships are loosely cohesive.

Figure 3. Reference architecture of Industry 4.0 based on standardized components and communication. The Industry 4.0 components can contain different objects (products or materials) and communicate using standardized interface. The picture is adopted from (Adolphs et al., 2015).



The standardized organizational frame bridges the physical and virtual word. The control of the process has been partially decentralized and decomposed to the respective components. Abstraction and encapsulation ensure that a particular machine can handle different products without the necessity to pre-program the machine. As long as the machine can handle the type of an Industry 4.0 component it can be processed based on the logic defined by the component itself.

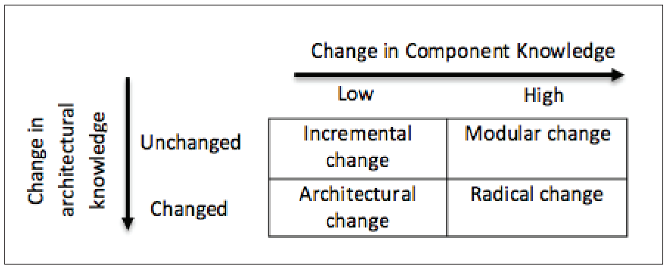
According to Adolphs et al. (2015) the preliminary reference architecture of Industry 4.0 distinguishes between type and instance. A type describes the concept from an idea and development phase to a first prototype of a product. This type serves as a template for the production stage. An instance of a type is a particular produced product (with a serial number). This conceptualization resonates with the class and object paradigm in object oriented development. A class provides a general template. Every logic which can handle this template can also handle any object of this class without knowing its specific instance. The reference model highlights that a similar behavior applies to the type construct, which shares information and data throughout the design and production phases and across different stakeholders, such as suppliers, engineers, and machinists.

Further improvement by implementing self-aware components, which can evolve and even adapt their problem solving behaviour and pass their successful logic on to others, would yield the analogy to inheritance in object oriented development (Lachmayer et al., 2014).

System change of Industry 4.0 and object-oriented development

The change from industrial automation systems to systems conforming with Industry 4.0 follows the categorization of Sircar et al. (2001), who applied the framework of Henderson (1992) to classify the change from structured development to object-oriented development. A system comprises of elements and architecture, reflecting their relationships. A change of a system can be anywhere among those two dimensions (cf. Figure 4).

Figure 4. Classification of the component and architectural change of a system (adopted from Henderson 1992)



Sircar et al. (2001) showed that the change during the analysis and design phase is architectural, because the relationship between elements undergoes a transformation. This is often discussed as a mind shift programmers have to assimilate (Due, 1993). During the implementation phase, the basic programming elements remained the same. Programming languages just supported object-oriented programming by adding more features and constructs. At later stages languages, such as Java, were designed to force programmers to develop in an object-oriented way. However, because the set of elements which constitutes a program did not change, the authors classified it as an incremental change during the implementation phase.

Conceptually, the elements of a production process such as machines and products undergo minor changes compared to the processes. Despite all the challenges to standardize a digital factory and its components (IEC 62832), the change is incremental, because the elements that constitute the whole remain the same. They are encapsulated in Industry 4.0 components, which enrich their functionalities. However, new elements that are elements of the production system are not added. From the architectural perspective, Industry 4.0 facilitates a shift from a centralized to a decentralized production system based on new relationships between those elements. Specifically, process logic is attached to an Industry 4.0 component rather than the machine. The communication among Industry 4.0 components, machines and humans, relies on context-aware systems rather than on centralized control of production. Context-awareness thus facilitates a self-controlled autonomous system.

Programming and manufacturing are inherently two different things. Software solves problems by creating and processing virtual data while manufacturing creates or processes physical products. Nevertheless, the concept of Industry 4.0 and its determinants enables similar change patterns by moving from structured production to object-oriented production in Industry 4.0. The revolutionary character could not only materialize in economic terms, but it could also manifest in a new way of how products are designed and manufactured. This materializes in new development methods and different skills for employees in the design and production phase.

In software development, the development process changes from the top-down waterfall model of structured development to the spiral model of object-oriented development. A similar change has been postulated by Kagermann and Wahlster (2013) for Industry 4.0 with a shift from structured development and production toward a continuous engineering process throughout the whole value chain and network.

Regarding labour skills, Hartmann and Bovenschulte (2013) visualize a skill roadmap for Industry 4.0. They highlight that the existing workforce will be complemented by new IT and technical skills such as Industrial ICT Specialists or Mechatronics Specialists. Based on their education, they already have a head start in object-oriented development, since it is part of any IT related educational curriculum. Thramboulidis (2015) proposes a cyber-physical system for industry automation and emphasizes the challenges for engineers to develop in an object-oriented way. The requirements in Industry 4.0 increasingly focus on cross-functional skills with the effect that the boundaries between blue-collar and white-collar workers will become more blurry (Kagermann and Wahlster, 2013).

As argued by Fichman and Kemerer (1997) complex organizational technologies in general and software process innovation, such as object-oriented programming, in particular create knowledge barriers that inhibit their diffusion. The authors emphasize that organizations with "higher leaning-related scale [scale of activities over which learning costs can be spread], greater related knowledge [existing knowledge related to focal innovation], and greater diversity of knowledge and activities be more prone to innovate, because such organizations can better amortize learning costs, can more easily acquire the knowledge needed to innovate" (Fichman and Kemerer, 1997). As Industry 4.0 carries a similar burden of organizational change, further research on Industry 4.0 assimilation and diffusion should not only focus on determinants and diffusion patterns, but also draw a comparison to object-oriented development diffusion.

Conclusion

Industry 4.0 induces a system change in manufacturing from central to decentral control of production. The change from a top-down to a bottom-up approach shares some similarities with the evolution from structured to object-oriented software development. First, the cause of the change is a system, either software or production, which gets increasingly complex and can only be handled by decentralizing the control. Second, both approaches start from the bottom and decompose the software or production logic to software objects or Industry 4.0 components, which encapsulate data and logic in single entities and are loosely cohesive in communication. Third, the distinction between class and object in software development and type and instance in Industry 4.0 facilitates a change in the development process from a top-down waterfall model to a spiral model in object-oriented development or continuous engineering along the value chain and network. Employing Henderson's framework (Henderson, 1992) the change pattern in Industry 4.0 resembles the change pattern in object-oriented development: incremental in the production phase and architectural in the design phase. In object-oriented development, architectural change induces a mind-set shift in designing software. By the same token, Industry 4.0 induces the development of new cross-functional labour skills. The delineation of blue-collar and white-collar workers will become less relevant.

References

Abele, E., Reinhart, G., (2011), Zukunft der Produktion: Herausforderungen, Forschungsfelder, Chancen. Hanser, Munich.

Adolphs, P., Bedenbender, H., Dirzus, D., Ehlich, M., Epple, U., Hankel, M., Heidel, R., Hoffmeister, M., Huhle, H., Kärcher, B., Koziolek, H., Pichler, R., Pollmeier, S., Schewe, F., Walter, A., Waser, B., Wollschlaeger, M., (2015). Statusreport Referenzarchitekturmodell Industrie 4.0 (RAMI4.0), 32 pp.

Bauer, W., Schlund, S., Marrenbach, D., Ganschar, O., (2014). Industrie 4.0 - Volkswirtschaftliches Potenzial für Deutschland, 46 pp.

Bauernhansl, T., Hompel, M. ten, Vogel-Heuser, B. (Eds.), (2014). Industrie 4.0 in Produktion, Automatisierung und Logistik. Springer Fachmedien Wiesbaden, Wiesbaden, 9 pp.

Bermann, S., Leonelli, N., Marshall, A., (2013). Digital reinvention: Preparing for a very different tomorrow, 24 pp.

Bitter, Rick et al, (2001). Object-Oriented Programming in LabVIEW, in: Bitter, R., Mohiuddin,, T., Nawrocki, M. (Eds.), LabView Advanced Programming Techniques. CRC Press LLC.

BMBF, (2013). Zukunftsprojekt Industrie 4.0. http://www.bmbf.de/de/9072.php.

Booch, G., (1986). Object-oriented development. IEEE Transactions on Software Engineering SE-12 (2).

Capretz, L.F., (2003). A brief history of the object-oriented approach. ACM SIGSOFT Software Engineering Notes 28 (2), 6.

Dahl, O.-J., (1981). Transcript of discussant's remarks, in: Wexelblat, R.L. (Ed.), History of programming languages I, pp. 488–490.

Dezhina, I., Ponomarev, A., Frolov, A., (2015). Advanced Manufacturing Technologies in Russia: Outlines of a New Policy. Foresight-Russia 9 (1), 20–31.

Due, R., (1993). Object-oriented Technology - The Economics of a new Paradigm. Information Systems Management 10 (3), 69–74.

European Commission, (2013). Factories of the Future, 136 pp.

Evans, P., Annunziata, M., (2012). Industrial Internet: Pushing the boundaries of Minds and Machines, 37 pp.

Fichman, R.G., Kemerer, C.F., (1997). The assimilation of software process innovations: An organizational learning perspective. Management Science 43 (10), 1345–1363.

Hartmann, E., Bovenschulte, M., (2013). Skills Needs Analysis for "Industry 4.0" based on Roadmaps for Smart Systems, in:, Using Technology Foresights for Identifying Future Skills Needs. Global Workshop Proceedings, pp. 24–36.

Henderson, R., (1992). Technological Change and The Management of Architectural Knowledge, in: Kochan, T., Useem, M. (Eds.), Transforming Organizations. Oxford University Press, New York.

Kagermann, H., Wahlster, W., (2013). Umsetzungsempfehlung für das Zukunftsproject Industrie 4.0, 116 pp.

Lachmayer, R., Mozgova, I., Reimche, W., Colditz, F., Mroz, G., Gottwald, P., (2014). Technical Inheritance: A Concept to Adapt the Evolution of Nature to Product Engineering. Procedia Technology 15, 178–187.

Lucke, D., Constantinescu, C., Westkämper, E., (2008). Smart factory - a step towards the next generation of manufacturing, in: Manufacturing systems and technologies: the 41st CIRP conference on manufacturing systems, pp. 115–118.

McGaughey, R.E., Snyder, C.A., (1994). The obstacles to successful CIM. International Journal of Production Economics 37 (2-3), 247–258.

NCST, (2012). A national strategy plan for advanced manufacutring, 51 pp. Accessed 4 July 2015.

PCAST, (2011). Report to the president on ensuring American leadership in advanced manufacturing, 56 pp.

Pichler, Reinhold, (2015). Positionspapier „AUTONOMIK und Standardisierung - vom erkannten Bedarf zur strategischen Ausrichtung", 10 pp.

Selvi, T., Chu, X., Buyya, R., (2009). Object Oriented Programming with Java: Essentials and Applications. TBS.

Sircar, S., Nerur, S.P., Mahapatra, R., (2001). Revolution or Evolution? A Comparison of Object-Oriented and Structured Systems Development Methods. MIS Quarterly 25 (4), 457–471.

Spath, D., Ganschar, O., Hämmerle, M., Krause, T., Schlund, S., Stefan, G., (2013). Produktionsarbeit der Zukunfst - Industrie 4.0.

State Council, (2015). State Council on the issuance of "Made in China 2025": Guo Fa No. 28, 35 pp.

Thramboulidis, K., (2015). A cyber–physical system-based approach for industrial automation systems. Computers in Industry 72, 92–102.

Vogel-Heuser, B., (2014). Herausforderungen und Anforderungen aus Sicht der IT und der Automatisierungstechnik, in: Bauernhansl, T., Hompel, M. ten, Vogel-Heuser, B. (Eds.), Industrie 4.0 in Produktion, Automatisierung und Logistik. Springer Fachmedien Wiesbaden, Wiesbaden.

Wahlster, W., (2014). Normung und Standardisierung - Schlüssel zum Erfolg von Industrie 4.0, 2014, Berlin.

Warnecke, H.-J., (1995). Aufbruch zum Fraktalen Unternehmen: Praxisbeispiele für neues Denken und Handeln. Springer, Berlin.

Weiser, M., (1991). The Computer for the 21st Century. Scientific American, 94–110.

Zuehlke, D., (2010). SmartFactory - towards a factory of things. Annual Reviews in Control 34 (1), 129–138.