

Towards combining layout and process models for mixed assembly facilities

Sven Stumm, Alwin Hoffmann, Hella Seebach, Bernd Kuhlenkötter, Wolfgang Reif

Angaben zur Veröffentlichung / Publication details:

Stumm, Sven, Alwin Hoffmann, Hella Seebach, Bernd Kuhlenkötter, and Wolfgang Reif. 2014. "Towards combining layout and process models for mixed assembly facilities." In Automation 2014: 15. Branchentreff der Mess- und Automatisierungstechnik; Smart X - powered by automation; Kongresshaus Baden-Baden, 01. und 02. Juli 2014, edited by Peter Adolphs, 277-88. Düsseldorf: VDI-Verlag.

Konzept erweiterter Modellierung zur Planung hybrider Montageanlagen

Towards combining layout and process models for mixed assembly facilities

S. Stumm, Univ.-Prof. Dr.-Ing. **B. Kuhlenkötter**,
TU Dortmund (IPS), Dortmund;
A. Hoffmann, Dr. **H. Seebach**,
Univ.-Prof. Dr.-Ing. **W. Reif**,
Universität Augsburg (ISSE), Augsburg

Kurzfassung

Kürzer werdende Produktzyklen sowie der Wunsch nach größerer Produktindividualität verstärken die Forderung nach einer hybriden Montage, die eine Kombination der Effizienz automatisierter Anlagen mit der Flexibilität manueller Arbeitsplätze ermöglicht. Eine erfolgreiche Realisierung der Bestrebungen erfordert eine Optimierung des Entwicklungsprozesses. Ein wesentlicher Grund hierfür ist, dass bereits bei der Erstellung des Anlagenkonzeptes eine Vielzahl von Anpassungen vorgenommen wird. Speziell im Bereich der industriellen Montage erfordert die produktive Nutzung hybrider Anlagen kontinuierliche Modifikationen auch nach der Inbetriebnahme.

Die meisten Änderungen an Anlagen werden jedoch nicht ausreichend dokumentiert und ein Rückfluss von Änderungsinformationen in ein ursprüngliches Anlagenmodell ist nicht sichergestellt. Dies ist jedoch essentiell bei weiteren Anpassungen der Anlage, da eine umfassende Nutzung aller vorhandenen Ressourcen und ein effizientes Design nur sichergestellt werden kann, wenn der aktuelle Zustand der Anlage vollständig erfasst ist.

Im Rahmen der Veröffentlichung wird ein Konzept zur Schaffung eines anpassungsfähigen Modells für hybride Montageanlagen präsentiert, welches eine fortlaufende Dokumentation und durch eine gezielte Erfassung interner Abhängigkeiten eine Modifikationsverfolgung ermöglicht. Dazu erlaubt das Modell die Spezifikation von Verknüpfungen zwischen Modell-elementen. Darauf aufbauend kann eine Analyse dieser Verknüpfungen die Elemente ermitteln, die von einer Modifikation betroffen sind.

Darüber hinaus werden Möglichkeiten zur Integration dieses Modells in bereits vorhandene Engineering-Tools sowie zur umfassenden Abdeckung von Anlagen- und Ablaufplanungen für Montageprozesse erörtert. Das Konzept basiert hierbei auf einer Kombination existierender Beschreibungsstandards: für die Anlagenplanung über die Automation Markup Language (AutomationML) [1] und zur System- und Prozessbeschreibung mit Elementen der Systems Modeling Language (SysML) [2]. Dabei greift dieser Ansatz zurück auf die funktionale Abbildung von Anlagenkomponenten in der Form von AutomationML-basierten SmartComponents und Elementen der kontinuierlichen Datenintegration, wie sie durch die *conexing* Lösung [3] vorangetrieben werden.

Abstract

Automation systems in general as well as assembly automation in particular continue to gain importance in current research and development. The main reason for this is the increasing demand in product variety combined with low and fluctuating product quantities while requiring an efficient use of the available resources. To achieve the necessary flexibility in production, mixed production environments are essential because they are able to combine the speed of automation with the flexibility of manual labor. Currently design and planning of mixed production environments leaves a lot to be desired. In most cases automated production and manual production are planned separately and are combined in a later step. Moreover, such facilities are continuously modified and improved during commissioning. However, most changes to the systems are not documented properly and, hence, the flow of information back to the original model after a modification of the facility is uncertain. This back propagation is essential, because any further development requires the current state of the facility for efficiently redesign and modify the assembly facility. The presented concept proposes extensions of current layout models to support the design and development of mixed assembly environments. The goal is to create a dynamic model of the production environment as well as internal dependencies, which simplifies documentation and modification tracking. For these reasons the proposed model combines elements of the Automation Markup Language (AutomationML) [1] for a functional representation of the environment layout and Systems Modeling Language (SysML) [2] for the design of the process flow.

1. Introduction

Throughout the life-cycle of a mixed production environment, many modifications are made. Especially in the area of assembly, facilities are adapted even after commissioning. Because of shorter version release cycles and the demand for more product variety, production environments need to be adjusted almost constantly [4]. However, implemented changes are

rarely documented sufficiently and modification information is often not propagated back into the original model. As a consequence, further development is prone to error, if it is directly dependent on this information. Investing a high amount of time for troubleshooting and redevelopment is the consequence. This paper proposes a novel model-driven approach which allows for a simple acquisition of all available data concerning mixed production environments as well as their dependencies and requirements. This is necessary to enable the future development of a partially automated modification-impact-analysis. To accomplish this, we have to take a further look at the current development process. Facility planning is currently done on two different but closely intertwined levels. One level is comprised of the layout planning, the other contains the planning of the process flow. The process flow is initially planned by creating an assembly priority graph with a subsequent detailed planning of the assembly instructions. Whereas part supply, positioning of facility components and therefore transfer times hinge largely on the layout planning. Both planning levels are mostly done separately, although an interactive combination of both planning processes right from the start could be useful. In spite of the fact that the production time and efficiency of the facility depends on both levels of planning, in most cases they are firstly integrated into an overall concept at the virtual commissioning or commissioning phase. By creating a common model for process and layout planning as well as modeling production dependencies and requirements a new way of interdisciplinary planning is made possible. Therefore the next section will explain current elements of assembly planning as well as the potential data models in the scenario of assembly planning. Afterwards the proposed combined model approach is explained as well as possible extensions of current assembly planning employing the model. In conclusion further usage and possible future work using the combined model approach is presented.

2. Assembly plant life cycle management

A lot of research and development has been done in the area of production planning, ranging from layout planning to production process and flow planning [5]. The concept presented in this paper extends current concepts using a model based approach for a combination of layout, process and functional planning. The research project *conexing* [3] is used as a basis for this approach. The *conexing* solution is an instrument for interdisciplinary planning and product specific virtual optimization of automated production systems. The ambition of the project is to connect every expert involved in the overall process interdisciplinary and across companies from the design phase up to the virtual production review in one common working environment. The common data structure is based on the Automation Markup Language

which is a data exchange format for plant engineering information based on the Extensible Markup Language (XML). AutomationML allows for a description of plant topology as well as geometry, kinematics and also logic programming. By extending these possibilities for automated production systems the connecting environment allows for virtual representations of functional entities, known as SmartComponents. These SmartComponents are used for a simulative depiction of real world production implements and can be used for virtual factory planning [6]. In the next section, we will discuss the proposed concept for assembly planning.

3. Concept

The main goal is the creation of a common model for process and layout planning. Therefore we look at possibilities to combine all necessary information for the planning of partially automated production systems into a model of the facility. For a complete coverage of the different planning areas we first classify different sets of elements and their representations necessary for the production system model. One set contains necessary mechatronic devices or rather production implements, which are responsible for executing production steps. For virtual factory planning, these implements require a virtual representation for simulating its real behavior. Another set of elements that need to be included in the model are the production steps or rather the different tasks required for production. These tasks are usually designed to meet different requirements, which should also be modelled. After discussing different modelling elements in the next sections, we will describe approaches to bring them into a common context by modelling internal dependencies. We will illustrate the overall approach using one common example of a simple generalized mixed assembly line (see *Figures 2-4*).

3.1. Implement modelling

A wide range of data formats can be used for three dimensional computer-aided design (CAD) data as well as product parameter descriptions. One notable data format is defined by a joined approach of Siemens PLM and the ProSTEP iViP association, which combines STEP application protocols for parameter description and the Jupiter Tessellation (JT) data format for visualization of geometries. A common alternative is the AutomationML data format, which is an open XML-based standard. To cover different domains in the field of automation, the AutomationML data format combines existing XML-based standards. This approach makes AutomationML easily extendable, which can be utilized for the necessary extension to realize the planned concept.

The current definition of AutomationML is based on the computer-aided engineering exchange (CAEX) format and employs it for modelling plant engineering data and production

topologies. Within AutomationML, computer-aided-design, kinematics and physical properties of objects can be described using the COLLABorative Design Activity (COLLADA) format, as well as MathML for mathematical contexts and relations. Descriptions of controller programming is integrated using the XML-based PLCOpen standard. By integrating these standards into the AutomationML format, the description of production implements, environments and plants is possible. Within the *conexing* project at the Institute for Production Systems (IPS) at the TU Dortmund University, this modelling of implements is extended to create virtual descriptions of automation components including their logical behavior. *SmartComponents* are created which are intelligent virtual representations of real components.

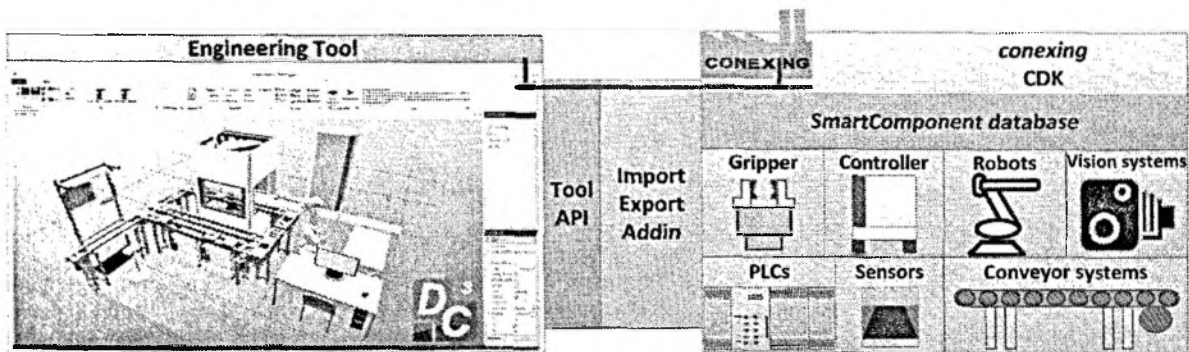


Fig. 1: Possibility to integrate the conexing Development Kit using a tool specific API interface for data conversion to implement import and export functionalities.

The *SmartComponent* data structure, as well as the *conexing* infrastructure is explained by Schyja et al. in [6]. *SmartComponents* can be employed to achieve detailed descriptions of production resources including behavior descriptions for mechatronic devices that can be integrated into simulation. In order to use this format for data exchange and to create an easy integration into different engineering tools, a software framework is developed. Using the *conexing* Development Kit (CDK) software tools are enabled to create and exchange *SmartComponents*. Figure 1 illustrates the possibilities to integrate the CDK into different engineering tools for import and export of production implement representation. These virtual production implements that correspond to real components can be used to create a realistic model of the production environment.

This allows a realistic representation of all required mechatronic elements within a mixed production environment. However, extensions for human modelling are necessary to improve the planning of mixed assembly lines. To illustrate the approach, an application scenario and a subset of necessary implement descriptions are given in Figure 2. It outlines a pictographic depiction of a possible layout for a mixed assembly line. For a better overview the compo-

ment descriptions are only given in the form of their roles within the assembly line not as specific component representations.

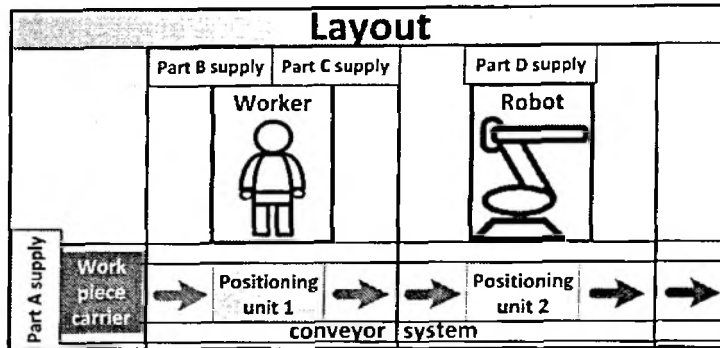


Fig. 2: Draft of a mixed assembly line layout. Depicted mechatronic components can be represented using conexing SmartComponents.

3.2. Task modelling

Although the described SmartComponents include logical behavior description of real automation components, modelling process operation sequences require further extensions to the existing framework. Process descriptions are currently not part of the existing AutomationML standard. Modelling tasks allow for a better dynamic planning of production flow as well as division of tasks between implements. Especially in the field of assembly, detailed descriptions of operation sequences are already available. Most descriptions currently exist in the form of text based assembly instruction manuals without a semantic connected data model, which prevents the creation of comprehensive models of assembly lines. In earlier stages of the assembly planning process, a higher level of abstraction is used and visual assembly plans are created. These assembly plans use symbolic representations for each assembly operation. For the presented example, the VDI 2860 standard [7] defines consistent symbols for assembly operations. Figure 3 shows an exemplary depiction of an assembly plan using VDI 2860 for the mixed assembly line shown before in Figure 2.

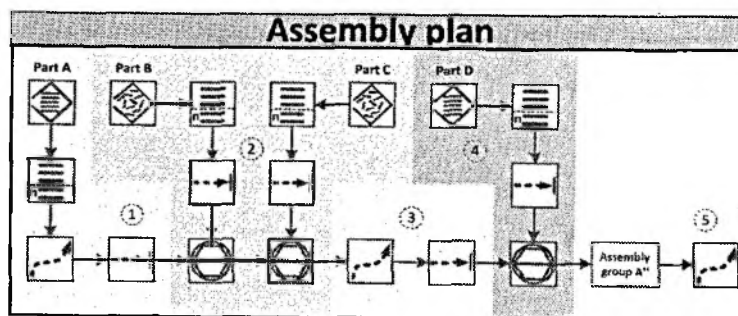


Fig. 3: Draft of an assembly plan based on VDI 2860. Sets of assembly operations are highlighted and numbered for later reference

Lotter and Wiendahl [8] differentiate between the assembly operations Joining (DIN 8593), Handling (VDI 2860), Inspection (VDI 2860), Adjusting (DIN 8580) and Special Operations (i.e. part processing operations). Each typical assembly operation can be represented with a symbol. A number of product life cycle management software tools also allow a digital representation of assembly operations. These digital representations consist of a description of assembly parts as well as the necessary operations for assembly. For a specific product assembly, this can be represented by a sequence of product states. However, within previous development steps of assembly lines the product assembly is analyzed to create an operation dependency plan. This plan is later used to define a specific assembly order and divide complex tasks into subtasks. For flexible assembly planning, these different degrees of detail should be covered by the desired approach. Hence, it is necessary to look at the similarities between these different forms of assembly plans. Comparing assembly plans within the different development phases from abstract assembly order to concrete assembly operations the following important common elements can be detected:

- The state of each assembly part before and after an assembly operation (State).
- Assembly operations results in a state change of an assembly part (Transition).
- Every assembly operation is bound by assembly conditions (Condition).

These are all typical elements of a state machines as defined by the Unified Modelling Languages [9] (UML) and incorporated into SysML. The hierarchical nature of state machines allows for a successive increase in the degree of detail throughout the development, within a semantically connected model of the assembly plan.

3.3. Requirement modelling

Supporting change is a well-known challenge for software development, because the maintainability of a system decreases significantly due to the lack of valid information about the current state of the system. The main reason is that knowledge over the system is getting lost continuously, because it is never or poorly documented, or original system designers are no longer available. This lack of valid information has manifold impacts. Especially changes to original requirements, which occur over time, cannot be traced and often there is no link between requirements and system components satisfying them. Moreover, constraints of the system, design decisions and the rationale behind them are lost. This makes it difficult or even impossible to reliably predict the impact of changes made to the system.

Hence, it is very important to analyze the system's functional and non-functional requirements and to document them in a formal and unambiguous way. A functional requirement

specifies an operation or a workflow that a mixed assembly system must perform. In the domain of assembly, examples for nonfunctional requirements include performance requirements (assembled pieces per hour) or assembly tolerances. Also legal issues or physical constraints are possible non-functional requirements. It is important to systematically capture and specify the possible requirement types for mixed assembly system.

To facilitate this, SysML introduces in addition to UML the requirement diagram which allows for modelling text-based requirements and their relationship to other requirements and to other model elements [2]. Each requirement in SysML has a name, a unique identifier and a text property for describing the requirement. Requirements can be decomposed into sub-requirements (i.e. a *containment relationship*) or they can be derived from other requirements. A *derive relationship* is always based on an analysis and the rationale should be documented. Furthermore, requirements can be related to other model elements. The *satisfy relationship* is used to assert that a system part satisfies a particular requirement. Using the *verify relationship*, a test case can be added that verifies that a particular requirement is satisfied.

However, requirements in SysML are only text-based and must be categorized and extended with further properties to formally analyze it. For that purpose, SysML allows to derive new requirement types with their relevant properties. We plan to use this feature to create a SysML extension that includes all requirement types relevant for mixed assembly facilities. By additionally creating new relationships between requirements or by extending existing relationship, it will be possible to formally document the system's requirements and to connect them to the relevant system parts.

3.4. Dependency modelling

The goal is to create an integrated model which comprises all relevant information and interconnects this information. Without such a model integrating all information during a system's life cycle, it is likely (and typically the case in practice) that sooner or later the cohesion between the system and its description is lost. Hence, the system becomes unmaintainable. To avoid this, it is necessary to support the documentation of an automation system's entire life cycle [10].

The main idea is that the engineer starts with an abstract description of the cell under construction. First, the model captures almost only the abstract requirements of the cell. The functional requirements can contain brief descriptions of the tasks or processes that need to be performed. However, also nonfunctional requirements such as performance, legal issues or failure tolerances are included. The information is refined adding more and more details.

The precise cell infrastructure including its implements is added (cf. Sect. 3.1) and the tasks that are performed are described (cf. Sect. 3.2). Furthermore, requirements will be refined and linked with model elements satisfying them. Therefore, the mechanisms available in SysML are used (cf. Sect. 3.3) and extended to match the domain. These links are an essential part of the model and important for further development of the cell.

Task-requirement dependencies are necessary to validate that all functional requirements are met. The functional requirements are refined and split into sub-requirement documenting the rationale for it. Depending on the formalism used to model tasks, the resulting functional requirements match single assembly operations. However, it is important to mention that the refinement process is essential for systems engineering. By modelling the relationship among requirements and between requirements and tasks, it is possible to understand design decisions and rationales. Fig. 4 shows some of the functional and non-functional requirements of the presented assembly cell example. The shown five functional requirements are described briefly and informal. However, each functional requirement is mapped into a set of assembly operations as Fig. 3 shows. Some requirements (especially timing constraints) span across a group of assembly operations. Hence, hierarchical grouping of assembly operations is necessary for *task-requirement dependencies*.

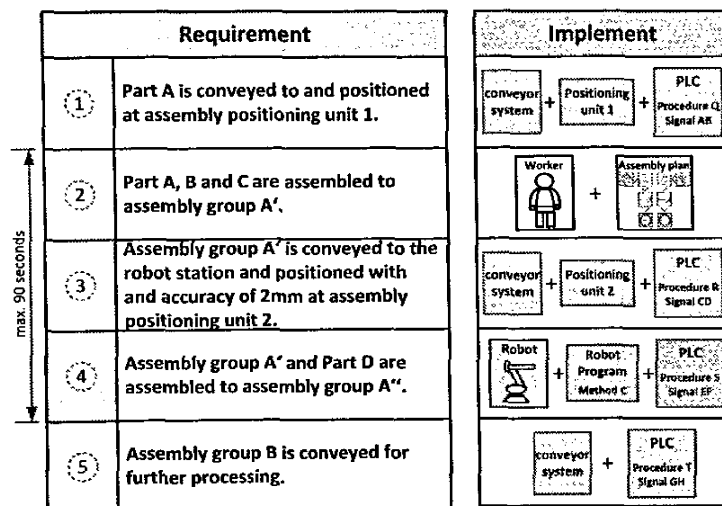


Fig. 4: Examples of dependencies between tasks requirements and implements.

Besides the links between requirements and tasks, single assembly operations must be linked to implements. These *task-implement dependencies* are necessary to validate that every task or assembly operation that is modeled has a counterpart that will perform the task. Hence, every assembly operation requires a set of implements. Besides modeling the devices, a comprehensive modelling includes the low-level behavior or programming as well as a

description of the behavior's cycle time and state with respect to input signals. Through a transitive relationship, *task-implement dependencies* ensure that every functional requirement is met by a corresponding implement. Fig. 4 shows these dependencies between requirements, tasks and implements.

It is worth mentioning that requirements, rationales and dependencies must not only be modelled during the initial development of the assembly facility but during the entire lifecycle. Hence, further requirements and relationships have to be added later, too. During virtual commissioning, for example, virtual device models are programmed offline in a graphical 3D simulation, because the real cell is not yet available. Additional requirements and rationales, which occur during off-line programming, can be documented in the model. Typical examples for such rationales are the introduction of additional synchronization states to avoid collisions. As off-line programming suffers from differences and inaccuracies between the simulation model of the cell and the real situation at the shop floor, motion programs must always be adapted during commissioning to match the real situation. These changes must be directly incorporated into the model. Typical examples for such requirements are restricted regions due to obstacles which are not present in the simulation (e.g. trailing cables).

4. Possibilities of dependency tracing

The traceability of requirements is an important part of the software development process and in particular of requirements engineering. In order to reconstruct the original requirements and to understand changes, links between different requirements and model elements on different levels of abstraction are essential [11].

By modeling *task requirement dependencies* and *task implement dependencies* as described in Sect. 3.4, it is possible to determine affected elements by later changing a requirement. Elements affected by such a change can be other requirements, which are in a relationship to the changed requirement, or tasks, which fulfill the changed requirement. Implements can be affected by changed requirements, too. They are either directly affected by a changed requirement or transitively because a related task was affected. Hence, these dependencies and relationships between requirements and other model elements facilitate the change of assembly systems throughout their lifecycle.

Assuming, there is a design modification in part D of the above presented example. This leads to a change in functional requirement 4 of Fig. 4 because the assembly group A' and part D need to be assembled in a different way. This changed requirement can lead to changes in related requirements. For example, requirements concerning the storage of part D or grasping part D are affected. Moreover, the task respectively the assembly opera-

tion fulfilling functional requirement 4 are affected (cf. Fig. 3). By formally modelling these relationships, it is possible to determine the possible effects of this modification.

Besides these effects, it is also possible to determine affected implements. Fig. 4 shows that the robot, methods of its robot program and parts of the PLC (i.e. procedure S and signal EF) are affected. Based on these information, it is possible to review these methods and procedures and to change the assembly system in a focused and systematic way. However, more research is necessary to determine affected implements in a comprehensive manner.

5. Synopsis & Future Work

Based on current data models for production facility layout and design as well as system modelling we proposed a novel and combined approach. The presented concept of extending existing layout models for mixed assembly facilities consists of basic models describing production implements, assembly tasks and requirements. By capturing the dependencies between these model elements, a comprehensive over-all model of a complete assembly line can be achieved. Using this, any change made to the assembly line can be used to modify a previously create model. This allows for an automated analysis of the change made and possibly also for the identification of effected model elements.

Extending AutomationML in order to model processes and tasks is part of current research. The given concept requires the integration of SysML interfaces to AutomationML as well as the possibility to reference SmartComponent Implements within SysML. A modelling framework needs to be implemented for researching the coverage of elements in the assembly domain using the proposed concept. Necessary extensions to the combined as well as each individual modelling area need to be identified and realized. After creating a complete model of an exemplary assembly line, methods for back-propagation of development information into the model need to be analyzed and further possibilities for a modification-impact-analysis need to be researched.

6. Acknowledgment

The conexing project at the Institute for Production Systems (IPS) of the TU Dortmund University is sponsored by the Federal Ministry of Education and Research (BMBF) with the Project Management Agency Karlsruhe (PTKA) as promoter of project finances.

7. References

- [1] R. Drath (Ed.): *Datenaustausch in der Anlagenplanung mit AutomationML: Integration von CAEX, PLCopen XML und COLLADA*. VDI-Buch. Springer, 2010.
- [2] Object Management Group: *OMG Systems Modeling Language (OMG SysML™)*. 2012 <http://www.omg.org/spec/SysML/1.3/PDF/> (Stand 28.10.2013)
- [3] M. Bartelt, A. Schyja, B. Kuhlenkötter and T. Benkner: *Konzept und Möglichkeiten der virtuellen Anlagenplanung*. PRODUCTIVITY Management, (3):31-33, 2013.
- [4] W. Eversheim, G. Schuh: *Integrierte Produkt- und Prozessgestaltung*, Springer, 2005.
- [5] D. Patzelt, J. Schallow, J. Deuse: *Data Integration in Digital Manufacturing based on Application Protocols*. Computer Science and Information Technology (ICCSIT), 3rd IEEE International Conference on, 2010.
- [6] A. Schyja, M. Bartelt, B. Kuhlenkötter: *From Conception Phase up to Virtual Verification using AutomationML*. CATS, CIRP Conference on Assembly Systems and Technologies, 2014.
- [7] Verein Deutscher Ingenieure: *Montage- und Handhabungstechnik; Handhabungsfunktionen, Handhabungseinrichtungen; Begriffe, Definitionen Symbole*, VDI 2860, Beuth Verlag GmbH, 1990.
- [8] B. Lotter, H.P. Wiendahl: *Montage in der industriellen Produktion*, Springer Vieweg, 2012.
- [9] Object Management Group: *Unified Modeling Language™ (OMG UML), Superstructure*, Specification Version 2.2, Feb. 2009, <http://doc.omg.org/formal/2009-02-02.pdf> (Stand 17.04.2013)
- [10] H. Koziolk, R. Weiss, Z. Durdik, J. Stammel, and K. Krogmann: *Towards Software Sustainability Guidelines for Long-living Industrial Systems*. Proc. Software Engineering (Workshops), 47-58, 2011.
- [11] A. D. Lucia, F. Fasano, R. Oliveto, and G. Tortora: *Recovering traceability links in software artifact management systems using information retrieval methods*. ACM Transactions on Software Engineering Methodology, Vol. 16 (4), September 2007