Proceedings in MANUFACTURING SYSTEMS

Proceedings in Manufacturing Systems, Volume 8, Issue 3, 2013 ISSN 2067-9238

A MODEL-DRIVEN ENGINEERING APPROACH FOR PRODUCTION SYSTEMS ILLUSTRATED ON AN AUTOMOTIVE TEST CASE

Susanne ALTENDORFER[1,*], Helmut ZSIFKOVITS[2]

[1)] Ass.-Prof. Dipl.-Eng. (FH) Dr. techn., Montanuniversitaet Leoben, Industrial Logistics, Leoben, Austria
[2)] Univ.-Prof. Dr., Head of Institute, Montanuniversitaet Leoben, Industrial Logistics, Leoben, Austria

Abstract: *Traditional organizational structures typically impede lean and flexible structures to cope with changing requirements and create barriers in and between enterprises along the value chain. New approaches to managing intra- and inter-enterprise processes – ranging from product development to manufacturing and delivery to the customer – have to be found. Innovative product engineering methods like concurrent and virtual engineering are on the rise within industrial practice, especially in the automotive and aerospace industry. This paper analyses different engineering approaches and explains a model-based engineering (MBE) approach in detail. MBE is presented as powerful method for the virtualization of complex automation systems (so called test facility systems) in the automotive industry. Starting with an analysis of the current situation the paper sketches the new approach and explains the underlying software modeling. At the end it concludes with the results of the new approach and the advantages for the company's process chain in general and for the product development and manufacturing in particular. Virtual Engineering is here the key for streamlining the processes and improving quality and lead times in the manufacturing process.*

Key words: *material flow, information flow, manufacturing, product development, concurrent engineering, virtual engineering, workflow systems, process optimisation*.

1. INTRODUCTION

Growing numbers of competitors, changing market requirements and decreasing product life cycles cause companies to react and adapt the way they are doing business. Over the years, however, organizations with traditional structures happened to create business departments as silos: they became big and shoreless [13]. This leads to blockades and information silos. Such organizational structures typically impede lean and flexible structures to cope with changing requirements. Furthermore, these information silos also influence the product and production life cycle in a negative way. Therefore the usage of new approaches − particularly concurrent and virtual engineering − has strongly increased over the last years [19].

Nevertheless, one of the critical questions is how to realize concurrent engineering and how to integrate it into the company. The authors will illustrate a concurrent engineering approach while using virtual engineering (simulation) approach within an automotive industry use case to optimize the industrial processes.

2. ENGINEERING APPROACHES

Concurrent Engineering is a systematic, holistic approach to take the elements of the product life cycle as well as the related processes (product development, sales, manufacturing, delivery and support) into consideration [8]. Furthermore, it is a powerful approach to influence the processes and the workflow of the employees to a high impact [14]. The idea behind this approach is, as it has evolved out of the product development, to reduce the elapsed time required to bring a new product to the market (time to market). This can only be achieved if all areas involved in the new product work together. Thus errors and necessary design modifications can be discovered early in the design process when the project is still flexible. This significantly increases productivity and product quality [8].

Virtual engineering is defined as "integrating geometric models and related engineering tools such as analysis and simulation, optimisation and decision making tools, etc. within a computer generated environment that facilitates multidisciplinary and collaborative product realisation [24]. Virtual engineering shares many characteristics with software engineering, such as the ability to obtain many different results through different implementations.

The concept of virtual engineering - is to provide a user-cantered, first-person perspective that enables the engineers to interact with the designed product in a natural way (real-world perspective) of an engineered system (product, system) and provide users with a wide range of accessible tools [6]. In this paper we concentrate on the simulation. This requires an engineering analysis engine (engineering model) that includes the geometry, physics, and any quantitative or qualitative data from the real sys-

[*]Corresponding author: Franz-Josef Str. 18, 8700 Leoben, Austria
Tel.: +43 3842 402 6027
Fax: +43 3842 402 6022
E-mail addresses: *susanne.altendorfer@unileoben.ac.at* (S. Altendorfer), *helmut.zsifkovits@unileoben.ac.at* (H. Zsifkovits).

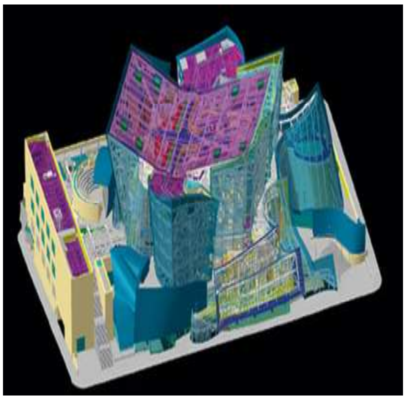

Fig. 1. BIM - Computer model of a building [21].

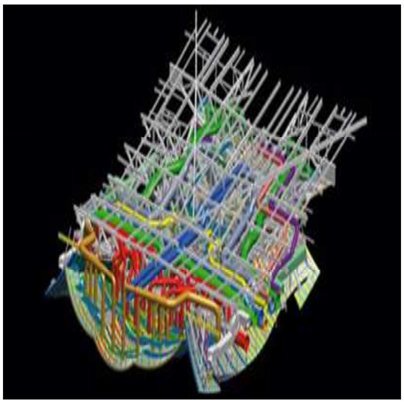

Fig. 2. BIM - Computer model with detailed plans of a particular subsystem [21].

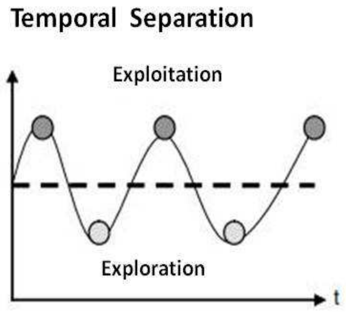

Fig. 3. Temporal separation in software development [16].

tem. The key components of such an environment include:

- User-cantered virtual reality visualisation techniques;
- Computer-Aided Manufacturing (CAM);
- Computer-Aided Engineering (CAE);
- Engineering decision support tools [6].

Thus, the user should be able to walk through the engineering model and observe how it works and how it responds to changes in design, operation, or any other engineering modification.

One example in the area of virtual engineering, which can also be classified as concurrent engineering, comes out of the construction industry: There the virtualization focused so far only on the visual look & feel of a building, including virtual walkthroughs. Eastman and others have championed an approach called Building Information Modeling (in the following abbreviated as BIM). The BIM approach is based on objects with their detailed properties [5]. Fig. 1 and Fig.2 show a BIM model for a building project with two different perspectives.

The model of the building has an associated database including information about the various relationships between these objects. Any changes in the model imply that all related objects are automatically updated. Thus, any mistakes are identified in the design phase. Quantities of material needed and hence construction costs can be calculated at a high accuracy at an early stage. Once the model has been created, detailed plans of particular subsystems can be extracted.

Thus, the mentioned engineering approaches - and the technologies behind - lead to the conclusion to introduce not only standalone software systems but business software to support business processes. In this context, Sundblad states that business software is often introduced for exactly one reason: it should support the business and its activities to increase the productivity and efficiency of the business [20]. The advantage of business software lies in the fact that business software, contrary to standalone software, can be integrated along the whole supply chain to get a higher scale effect.

3. CONSTRAINTS OF MONOLITHIC SOFTWARE

Many software systems are used over decades, with just slightly adapting the system. Business needs are not really considered in depth but only to a point where adaptations to existing systems are risky and dangerous. This fact is not a phenomenon within the automotive industry. This is quite a common phenomenon. Raisch describes this phenomenon as "*Ambidexterity*" [16]. This means that an organization differs between two cycles of software reengineering: (1) the Exploration and (2) the Exploitation phase. The former signifies a phase where new systems are tested and introduced. This phase, however, requires a high level of readiness to assume risks, as many unknown components are predominant in this phase. The latter concentrates on exploiting existing systems as long as possible. Here just small and almost insignificant adaptations are made at the software systems. This means no risks and no changes to existing workflows [16]. Fig. 3 illustrates how the two phases interchange with each other. Normally the exploration phase is significantly longer.

Companies wait extremely long, sometimes too long, before changing a system [23]. Over the years the requirements and the user experience have changed and the technological feasibilities enable a completely different approach to problem solving. This evolution makes it easier for the user to cope with the difficult tasks and complexity could even be reduced. Therefore it is necessary to break up with existing system and to start from scratch in order to change the current situation. Monolithic software architecture is often of great hindrance instead of supporting the business processes. It does not support the business requirements and as standalone software it even makes the processes more complex due to many interfaces needed. Streamlining business processes is therefore hardly possible without a major overhaul of the software architecture that forms the basis for supporting the business processes.

This shift to business software means a huge paradigm change for companies. It is often not that easy to step into such a radical change project as the results are not easily predictable and case studies on a long time horizon are not available. However, introducing business software would support companies in restructuring their business and making their processes more efficient [11]. The most well-known and widespread business software is ERP (Enterprise Resource Planning) systems [17]. However, business software is not only restricted to ERP systems though a concentration on those systems is recognizable, due to the fact, that ERP systems are very common no matter from which provider, and the integration cycle is documented and rather similar no matter what's the company's business [18]. Business software, however, can also be implemented for technical matters along the whole product life cycle. Thus a better process integration and support would be granted [7]. A big barrier in this area nowadays is that almost no examples of business software integration in the technical area exist.

The project described in the following chapters investigates the paradigm change to business software in the technical area – through virtual engineering - within the automotive industry. Especially in the automotive industry many technical software systems exist along the supply chain, which could be replaced by integrated business software, which could then have a high impact on the workflow of the business processes. It should be noted, that the paper contains a brief outline and short description of a prototypical example of the specific business software to support the business processes.

4. USED CASE: AUTOMOTIVE INDUSTRY

The automotive industry is characterized by a huge product portfolio and intensive development and production cycles as the products and systems are very complex and consist of many software-intensive products. The goal of the research together with our industrial partner has been to develop a framework for model-driven generation of automation systems, which radically simplifies the configuration and operation of test facility systems. This framework should then be the basis for business software to be implemented along the whole product life cycle. As a consequence, the associated business processes should be significantly streamlined.

A test facility system basically measures, records, and visualizes numerous values provided by sensors according to test plan. Typical test facilities consist of hundreds of thousands of components. According to Martyr "an engine test facility is a complex of machinery, instrumentation and support services, housed in a building adapted or built for its purpose. For such a facility to function correctly and cost-effectively, its many parts must be matched to each other while meeting the operational requirements of the user and being compliant with various regulations" [10].

4.1. Current Situation

To overcome the previously mentioned constraints of monolithic software new ways of engineering have to be found. The main intention for a new modeling approach is that it supports the domain experts (according to their needs and domain experience) in describing the corresponding product or system. An essential requirement is that the description can start with a coarse version, early in the processes, which can be refined in detail throughout a project. In order to facilitate custom-configured automation systems and at the same time reduce the development, deployment and maintenance efforts, we found that virtual engineering with a certain model-based software approach is required.

4.1.1. Monolithic Software Architecture. Automation systems need to be tailored to customer demands in a straight-forward way, which is neither well supported by the current monolithic software architecture nor does the software support the process chain [13, 20, 24]. The existing software has evolved over the past decades and comprises million lines of code, mainly written in C++ and C, which makes adjustments quite difficult. It requires error-prone editing of parameters (in the order of tens of thousands) in spreadsheet-like tables, as well as the adaptation of configuration files and scripts scattered in the file system. Thus the handling is time-consuming and expensive. Required changes, that would make the software systems flexible to business requirements, are neither possible with the existing software architecture nor is any potential effort for writing code for transforming the data justifiable. A software engineering has to be launched to make the system flexible to current requirements. Therefore it is highly recommended that the new tool supports the whole processes. A lot of information, which is nowadays entered into the tool in the manufacturing phase already have to be available in an earlier stage, e.g. in the sales phase. Therefore it is useful to rethink the product integration and use the software as business software along the whole process chain.

4.1.2. Current business process landscape. The current software is restricted to the manufacturing process without having any connection to the up- and downstream processes. Fig. 4 illustrates this situation. The handover points from one process to the other are without any tool support.

Thus, relevant information might be lost throughout the process chain and information needed in the progression of the project is not available as the importance of these data might not be clear in an earlier phase. Currently, each process uses different tools, like Excel sheets or proprietary tools. These gaps cause a significant information loss and extra manual conversion and transfer efforts.

4.2. Model-Driven Engineering Approach

To show the feasibility of any business software concept both sides, the business processes and the software engineering, have to be considered. It is crucial that a software system matches the requirements of an organization and that the software enables the business processes to adapt to business requirements. For the described research area, the overall complexity of test facility systems demands for business software that applies stepwise refinement. The aim is that a unified method and tool set is established, with which everybody throughout the process chain can interact. The resulting model of the overall test facility system should then correspond to the real-world test facility in all its relevant details.

The principal idea is to reduce the complexity through an object-oriented model which enables a graphical 1:1 representation of real-world items of the test facility system. Thus, this kind of modeling is referred to as deep virtualization of the corresponding real-world system. In the beginning a coarse-grained model can be defined which is incrementally refined along the value chain. Models are the main artifacts describing a system under test, and a model at a certain level of abstraction can be transformed into another model at a possibly different level of abstraction: the hypothesis is that

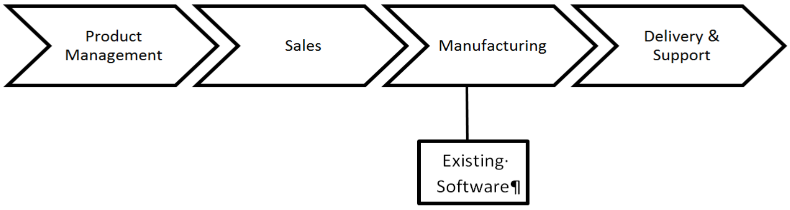

Fig. 4. Value chain with the current software system.

a software system that allows a stepwise modeling of a test facility automation system would support the overall business process chain and thus forms the basis for process streamlining and cost cutting.

By examining the examples of how to model automation systems, problems of conventional modeling languages such as UML were identified [12]. Instead the decision was made for what Atkinson and Kühne have called clabject-based modeling, an approach that unifies the notion of classes and objects [3, 4]. The advantage hereby is that the domain engineers do not have to concentrate on specific differences between instances, classes and objects. They can mainly concentrate on the engineering of the domain elements. Thus, this approach supports the stepwise refinement, as people from different departments can work with the tool without having to know the underlying software architecture.

The main intention for a new modeling approach is that it supports the domain experts (according to their needs and domain experience) in describing a test facility system. An essential requirement when describing a test facility system is that the description can start with a coarse version, early in the processes, which can be refined in detail throughout a project. Moreover different levels of detail and different views – specific software and hardware views – on the test facility system ease the work of the domain experts.

With the clabject-based approach a library of all relevant elements is developed. The element library is then relevant for the software usage. It is filled in the beginning of the value chain (in the product management) and then used throughout the processes. The modeling of the product library, however, is essential for the whole product management and can also enable a better product management and production planning concerning a reduction of the complexity of product variants.

4.2.1. Requirements for Model-Driven Engineering. The main requirements for a new software modeling language and tool describing test facility systems are formulated in

- *Requirement* 1: The new software modeling approach must allow a stepwise refineable description of an automation system. The key is to create automation system at various different levels of detail, starting with a coarse-grained description in the early processes and refining the system along the process chain.
- *Requirement* 2: To enable a stepwise refinement, a graphical representation of real-world items for automation systems has to be introduced. Thus the graphical representation enables the depiction of the automation systems in different views, e.g. hardware and software view.

Fulfilling these requirements, a model-driven engineering with clabject-based modeling approach can be used as integrated business software supporting the whole value chain as this is a promising approach for coping with the inherent complexity of large software-intensive systems such as automation systems.

4.2.2. Clabjects-uniform representation of classes and objects. In the context of domain models, Atkinson and Kühne define accidental complexity as introduced due to mismatches between the problem and the modeling means to represent the problem [3]. They argue that modeling languages that allow using only two levels, such as UML with the class and object level, induce accidental complexity when modeling domains that inherently require more modeling levels. The solution they offer is the concept of a clabject, a modeling entity that has a class facet as well as an object facet. Fig. 5 shows the clabject-based approach for the domain of test bed automation systems [1]. The notation used here is similar to that of the original clabject concept. Each model element has a compartment for the name, and a combined compartment for the type facet and the instance facet. The dashed arrows between the levels represent the "instance of" relationship. With a uniform representation of type facets and instance facets, our example can be modeled in a natural way.

By definition, the clabjects at the top-level only have a type facet, whereas the clabjects at the bottom level only have an instance facet [2].

5. RESULTS

5.1. Benefits for the process landscape

A typical business process chain covers aspects from Product Management, Sales, and Manufacturing to Delivery and associated roles on the tool. This clabject-based modeling approach can be integrated along the process chain with a real visualisation of the test facility system. Fig. 6 gives a first overview of the real test facility system modeling tool based in clabjects. This approach demonstrates the core benefit of such a modeling tool in Fig. 7. Through the integration as business software, indicated as a constant banner (referred to as Business Software in Fig.7), it supports all essential processes and it will have predefined interfaces to other tools, which are relevant for the processes. While the new business software is the technical backbone, other tools are still used for economic matters – like a sales tool or a calculation tool. All technical solution components from the business software can be automatically exported to other tools when needed. Thus, the workflow is well supported and needed information can be easily accessed at the right time. It can also be realized that between the processes the gaps disappeared.

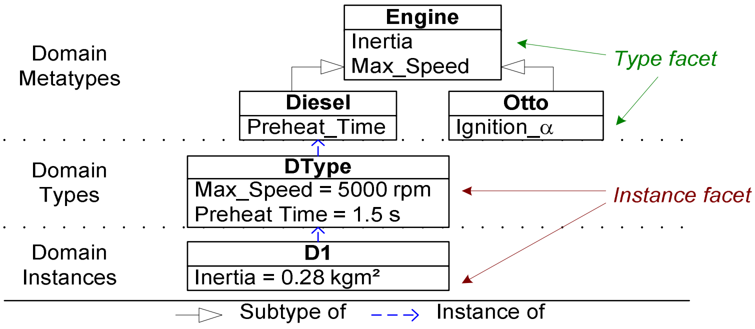

Fig. 5. Engine hierarchy with clabjects [2].

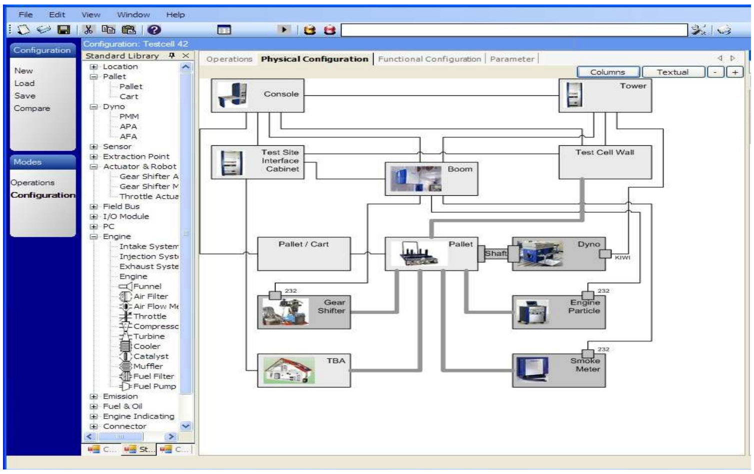

Fig. 6. Schematic real-world visualisation of a test facility system.

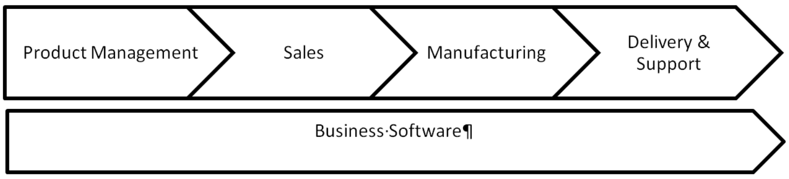

Fig. 7. Envisioned value chain supported by a business software.

Another effect of the integration of the business tool is that several monolithic standalone tools can also be completely replaced. Thus the software landscape will be cleaned up. Thus streamlining makes the work of the people easier, as fewer tools have to be used. On the other hand the reduction of tools is also significantly displayed in the cost structure of the IT department.

Realizing this clabject-based modeling approach also has manifold benefits for the whole workflow of an organisation, which are based on some well-known concepts: *Reusability*; *Correctness*; *Quality assurance*; *Visualisation* [15].

The application of these concepts has its individual benefits on the different processes within an organization. Therefore all relevant information should be added gradually to the test facility model in each process phase according to the particular requirements. Thus, the level of detail represented by the model increases over the project progression.

A further advantage is that errors or dependencies in the design and development of the systems can be identified already at an early stage in the sales phase, avoiding high costs due to late detection and fixing of such errors in the subsequent manufacturing.

The changes in the workflow also require a change in the role model of the existing processes. It is an advantage of the tool that it is not intended to be a tool for experts only. Employees working in all business areas should use the tool in their context. The granularity and level of detail of information however will be adapted to the role concept. Thus the business and software reengineering has also a high effect on the people within a company [9]. It is not only that they have to change what and how they do but the basic ways how they think is altering. Therefore any reengineering project should be accompanied by a special communication strategy within the company to prepare all people affected by the change.

5.2. Effects on the process cycle time.

In general, there are two relevant points where first savings can be achieved:

A reduction of cycle times can either be achieved

(1) with an elimination of process steps or

(2) with the shift of process steps to upstream processes.

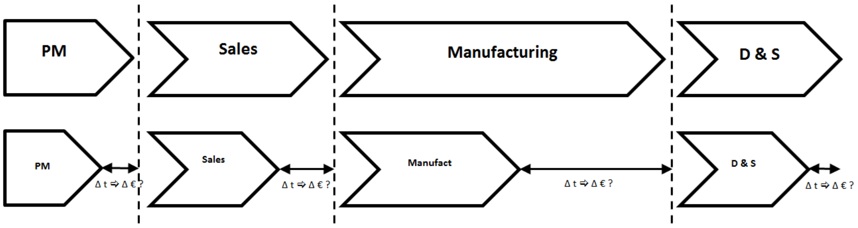

Fig. 8. Streamlined process chain after tool introduction.

The first one can be achieved as the system offers the possibility that process steps are done automatically. The latter one enables a cost reduction as several work steps can be done in house.

First analyses show that savings between ten and fourteen per cent can be achieved. This first analysis, however, is only based on scenarios where the business software usage is mainly concentrated on the manufacturing process. With a process-wide implementation, as shown in Fig. 8, even more savings can be achieved.

5.3. Effects on external and internal customers.

Furthermore, the integrated tool with its visualisation approach can also increase the customers' satisfaction, as the tool handling becomes easier. The satisfaction will also increase with internal customers. Training times for new employees can be radically shortened, which has a motivational effect on the employee and a positive financial effect for internal and external clients.

6. CONCLUSION

This paper demonstrated the importance and advantages of virtual engineering for the whole product life cycle. An appropriate modeling approach to support MDE was presented with a use case in the automotive industry. It showed that virtual engineering can support - with a process wide tool - a streamlining of the processes. A software prototype was developed and is currently tested. First analyses indicate that savings up to a two-digit percentage should be achieved.

REFERENCES

[1] T. Aschauer, G. Dauenhauer, W. Pree, *Towards a Generic Architecture for Multi-Level Modeling*, Proceedings of the 2009 Joint Working IEEE/IFIP Conference on Software Architecture & European Conference on Software Architecture WICSA/ECSA 2009, September 2009, pp 121-130. Cambridge. UK.

[2] T. Aschauer, G. Dauenhauer, W. Pree, *Multi-Level Modeling for Industrial Automation Systems*, 35th Euromicro Software Engineering and Advanced Applications (SEAA'09) Conference, August 2009, Patras, Greece.

[3] C. Atkinson, T. Kühne, *Model-Driven Development: A Metamodeling Foundation*, IEEE Software, Vol. 20. No. 5, pp. 36-41, 2003.

[4] C. Atkinson, T. Kühne, *Reducing accidental complexity in domain models*, Software and Systems Modeling, vol. 7, no. 3, 2007, pp. 345–359.

[5] C. Eastman, P. Teucholz, et al., *BIM Handbook. A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers, and Contractors*. Wiley & Sons Inc. New Jersey.2008.

[6] G. Huang, K. M. Bryden, *Interactive Design Using CFD and Virtual Engineering*, 10th AIAA/ISSMA Multidisciplinary Analysis and Optimization Conference, August - 1 September 2004, Albany, New York, 2004.

[7] J. Kawalek, *Rethinking Information Systems in Organizatons, Integrating Organisational Problem Solving*, Routledge Chapman & Hall, New York, 2008.

[8] A. Kusiak, *Concurrent Engineering: Automation, Tools, and Techniques*, John Wiley&Sons, 1998.

[9] K. Lewin, *Researching Organizational Values and Beliefs*, Praeger Frederick. 2001.

[10] A. Martyr, *Engine Testing: Theory and Practice*, Butterworth Heinemann, 2007.

[11] S. Muthum, L. Withman, *Business Process Reengineering: A consolidated methodology*, Proceedings of the 4th Annual International Conference on Industrial Engineering Theory, Applications and Practices, Texas, 1999.

[12] Object Management Group, Unified Modeling Language Infrastructure, Version 2.1.2, 2007.

[13] M. Osterloh, J. Frost, *Prozessmanagement als Kernkompetenz: Wie Sie Business Reengineering strategisch nutzen können* (Processmanagement as Core Competence: How to strategically use Business Reengineering), Gabler Verlag, Wiesbaden, 2006.

[14] J.P. Pennell, R.I. Winner, R.I., *Concurrent engineering: practices and prospects*, Global Telecommunications Conference and Exhibition 'Communications Technology for the 1990s and Beyond', IEEE, pp.647-655, November 1989, IEEE, Dallas.

[15] W. Pree, *Design Patterns for Object-Oriented Software Development*, Addison-Wesley/ACM Press, 1995.

[16] S. Raisch, *Balanced Structures: Designing Organisations for Profitable Growth*, Long Range Planning, pp. 483-508, 2008.

[17] SAP, `http://www.sap.de,` accessed: 2013-05-17.

[18] W. Semar, Weiss + Appetito Holding AG - *Software für den ganzen Konzern* (Software for the whole company). In: Wölfle, Ralf; Schubert, Petra (Hg.): Dauerhafter Erfolg mit Business Software - 10 Jahre Fallstudien nach der eXperience Methodik. München: Carl Hanser, S. 51-64. 2009.

[19] M. Schumann, M. Schenk, U. Schmucker, G. Saake, *Digital Engineering- Herausforderungen, Ziele und Lösungsbeispiele* (Digital Engineering – Challenges, Goals and Solutions), 14. IFF-Wissenschaftstage, 8. Fachtagung Digitales Engineering und virtuelle Techniken, Michael Schenk (Ed.), pp. 191-200, Magdeburg, June 2011, Fraunhofer Verlag, Stuttgart.

[20] S. Sundblad, P.Sundblad, *Business Improvement through better architected software*, Microsoft Architect Journal. Journal 10, 2007.

[21] The Economist. *From blueprint to database*, `http://www.economist.com/` accessed: 2013-05-17.

[22] D. Ungar, R. B. Smith, *Self: The power of simplicity*, Proceedings of OOPSLA '87, ACM SIGPLAN Notices, Vol. 22, pp. 227–242, 1987.

[23] R. Winter, *Business Engineering*, Integrierte Informationssysteme. Springer Verlag Berlin, p.17-37, 2008.

[24] Xiao, A. and Bryden, K. M., *Virtual Engineering: A vision of the next-generation Product Realization Using Virtual Reality Technologies*, Proceedings of ASME Design Engineering Technical, Computers in Engineering Conference, 2004.