1 REQUIREMENTS MODELS FOR MODULAR PRODUCTS

Carsten Stechert[a] and Hans-Joachim Franke

Institute for Engineering Design, Technische Universität Braunschweig, Langer Kamp 8, 38106 Braunschweig, Germany. Tel: +49(0)531/391-3349, Fax: +49(0)531/391-4572. E-mail: [a] stechert@ikt.tu-bs.de

This paper addresses modeling of requirements in the field of high dynamic parallel robots. Parallel robots are mechatronic, mass customization products. Every task at each customer is unique. Thus, the need for a domain integrating method that supports the development of modular systems was the main motivation for this work. Requirements are seen as the core of a successful product development. All steps of the development process and every subsystem should support the initial goals. A requirements model on the basis of the Systems Modeling Language (SysML) is shown. It integrates, visualizes and documents several different model elements. A classification of requirements and relations (both qualitative and quantitative) helps detecting goal conflicts and estimating change impact. Assigning certainty and change probability to the elements gives directions how to decompose into modules and where to integrate change propagation blocker or flexible modules.

Keywords: Requirements, Design Methods, Modular Product, Parallel Robots.

1. INTRODUCTION

This work addresses modeling of requirements in the field of high dynamic parallel robots. Parallel robots are mechatronic, mass customization products. Every task at each customer is unique. Thus, the need for a domain integrating method that supports modular systems was the main motivation for this work.

The development of modular systems is often considered where a number of different variants or a certain degree of flexibility is needed, i.e. modular systems are not only efficient for mass products but also for so called mass customization products. But it is important to know the main cost drivers and other boundary conditions during the product lifecycle, thus module drivers that lead to the modularization strategy.

Requirements are the core of a successful product development. All steps of the development process and every subsystem should support the initial goals. However, development processes are in danger that the more concrete and thus more complex the product gets developers start to forget about the impulse for the development of the specific product. They get lost in the huge number of requirements that were generated during the development. Those are related to different modules and diverse domains.

A requirements model is shown that uses the Systems Modeling Language (SysML) as a basis. SysML is a widely known standard e.g. in software development, electronics, and automation. The extended model integrates several different model elements (e.g. scenarios, functions structure, mechanical structure, and manufacturing process). These elements are visualized and documented. Based on the developed classification of requirements and relations (both qualitative and quantitative) an analysis is possible that helps detecting goal conflicts and estimating change impact. Assigning certainty and change probability to the elements gives directions how to decompose into modules and where to integrate change propagation blocker or flexible modules.

2. DEVELOPING MODULAR PRODUCTS

The development of modular products is often an approach to save costs by economies of scale and to decrease the time-to-market for a number of variants. Hence, the number of internal variants is to be kept small, while the external variants should be high to fulfill additional customer wishes.[1] Drawbacks are increasing development risks and development time. In addition, the company intern processes and structures will be more complex.[2] Thus, it is important to know the specific aims of the modular system development and thence to define modules in a proper way.

Around 40 more or less different methods and methodologies to designing modular systems are summarized in Ref. 1. One common approach (e.g. Ref. 3) bases on three basic steps: (1) decomposition of the system into elements, (2) documentation of the interactions between the elements, and (3) clustering the elements into architectural and team chunks. In the conceptual design phase functions structures are modeled that contain function elements and flows between them (e.g. Refs. 4 and 5). In embodiment design more detailed components are built up as function-carriers and modules evolve. For this purpose Blees and Krause[6] developed the Module Interface Graph (MIG) that displays components with respect to their geometrical position while lines represent interfaces. A comparable approach is shown in Ref. 7 for parallel robots.

Ericcson and Erixon[8] suggest the Modular Function Deployment (MFD) as supporting method to generate an optimal modular product design with respect to actual company-internal boundary conditions. Design rules such as "Design-for-Manufacturing/Assembly" are taken into account to optimize the modules. An adapted set of module drivers all related to assembly aspects are presented in Ref. 6: work content, skills, lead time, automation, tools, separate testing, process, handling, storage. A broader approach is stated in Ref. 9 grouping different types of module drivers according to time, degree of commonality, product lifecycle and location of effort. According to Ref. 10 a system is simply characterized by a bigger number of relations between system elements within the system boundaries than outside them. Following up this approach, one has to distinguish different layers of subsystems within the same product, e.g. related to information flow, physical contact or organizational aspects like departmental borders.

The subsystems of a product are to be developed as independently as possible. But if developers do not distinguish between the different system layers and generate modules according to just one (intuitively found) layer, this might lead to suboptimal modular systems. One possible consequence is redundant structures, i.e. synergetic effects are not used and the same problem is solved twice, because module interfaces are generated just because of departmental or domain separation. Another consequence is that the development focuses on just one or two aspects of the product (e.g. manufacturing, assembly) and does not consider the really important module drivers (that might even be different in different subsystems). The requirements model describes early the real aims of the modularity and by analyzing the relations modules can be separated more purposefully.

3. MODELING THE PRODUCT

The development of a product makes use of a variety of different models or "partialmodels" that describe a specific view on the whole system. Two definitions of model are cited below:

"Models express relations between real conditions in an abstract form. As a copy of reality models own simplifications that on the one hand cause a loss of realness, but on the other hand bring transparency and controllability of real relations."[11]

"Model is a purpose-dependent, finite, simplified, but still adequate representation of whatever is modeled, allowing us to abstract from its unimportant properties and details and to concentrate only on the most specific and most important traits."[12]

Both definitions contain the aspects of abstraction and simplification to bring transparency and to concentrate on the most important characteristics. Ort[11] additionally states, that models bring controllability of real relations: We cannot control what we do not understand.

According to Avgoustinov[12] a model should represent the subject to be modeled, ignore unimportant details (abstraction) and allow a pragmatic usage. The purposes are to support and improve

the understanding of the matter and build a common basis for discussion and information exchange. Moreover, models should allow comparison of different solutions as well as analysis and prediction of behavior and characteristics of the system to be designed. The organization of a model should contain its structure and architecture. Furthermore, interactions between components, component interdependencies and important external relations should be taken into account.

One important aspect of a model is its representation, thence the visualization, of its contents. Salustri *et al.*[13] mention that "there is relatively little use of diagrammatic visualization of qualitative information in the early stages of designing", although designers are seen as "visual thinkers". Within this context two principles are stated[13]:

- "Simplicity is power."
- "Diagrams augment cognition."

A model of the designed product is always some kind of documentation, too. It documents the actual state of work, allows for discussions and gives a basis for presentations to e.g. stakeholders. The necessary degree of formalism depends on the project and its state. The more creativity is demanded the more would formalism hinder. Many product development processes thus suggest a from-rough-to-detailed approach. The model should just be as detailed as necessary to provide a commonly understandable basis for all persons who might get in touch with this model. Wherever more detailed aspects are needed a submodel for a subset of project members should be generated.

The Systems Modeling Language (SysML) is a notation that allows modeling a product on different levels of abstraction and with different viewpoints. It is a widely known notation within the fields of software development, electronic design, automation, and (in parts of) mechanical engineering. It uses parts of UML (Unified Modeling Language) and extends it to the needs of systems engineering. It is provided by several commercial and open source modeling tools. OMG SysML v1.0 was issued as Available Specification in September 2007[14] and provides a common basis, so a better exchangeability, to describe requirements, as well as structure (e.g. blocks, packages, constraints) and behavior (e.g. activities, use cases). All relations can be visualized in diagrams and formatted to desired views. Also lists (e.g. requirements lists) and matrices (e.g. to support MFD) can be generated.

The requirements model is the core that forces the development process to fulfill the initial customer needs, the companies' strategic aims, and other constraints, e.g. arising from laws and regulations. Thence, from this model the aims for the development of the modular system are generated, made transparent and can be traced covering different domains and abstraction levels. Figure 1 illustrates this thinking approach as a schematic overview for a parallel robot of type HEXA (6 DoF). In the following, four aspects of the requirements model are discussed: Surroundings, structure, relations, and analysis.

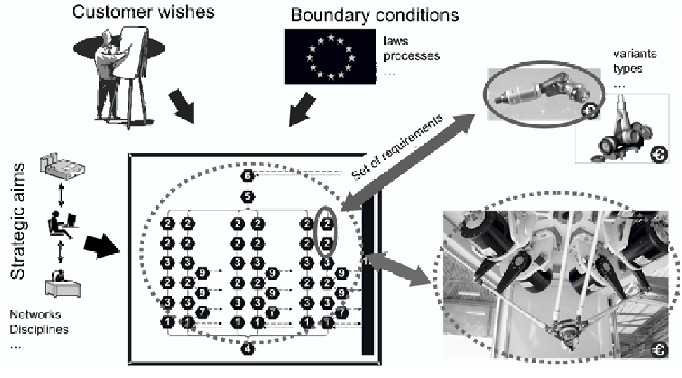

Figure 1. Requirements and constraints for the parallel robot Hexa (schematic overview).

3.1. Surroundings

One of the first steps in the development process is to analyze the product surroundings (e.g. Ref. 4). Here, the whole product lifecycle has to be taken into account including different scenarios (e.g. Ref. 15) or use cases, with all related actors, surrounding environment and possible disturbances. Furthermore, the different product views from different domains have to be considered and those requirements generated by later development steps (e.g. simulation, manufacturing) should be gathered, because they can contain module drivers, too. A systematic documentation helps to identify and to use the collected requirements and constraints.

3.2. Structure

For a better accessibility information should be structured. Requirements are structured in a hierarchy such as goal, target, system requirement, and subsystem requirement. Furthermore, they are allocated to a domain and to a purpose in the development process. Getting more concrete, requirements can be allocated to their concerning subsystems. In addition to well established attributes (wish/minimum/fixed, source) it makes sense to assign certainty and change probability. Hence, the more uncertain a requirement is the more carefully should a module be designed, i.e. the module should be detailed as late as possible. If due to interactions this is not possible it should be kept as flexible as feasible. For this purpose design rules (cp. Ref. 16) can be of importance.

3.3. Relations

Model elements are related to each other. In Ref. 17 a basic classification for relations was suggested according to development steps, granularity, support, direction, linking, and quantifiability. The systematic integration of relations into the model helps designers understanding the matter and be aware of interfaces to other disciplines. During synthesis steps, getting aware of relations might lead to new model elements. Jung[18] showed this for an interdisciplinary development in the field of medical apparatuses. In addition, the modeling of relations is the basis for the analysis of the model, discussed below.

3.4. Analysis

One important aspect during the development of complex products is to detect goal conflicts both in early qualitative and later quantitative phases. The earlier one gets aware of possible goal conflicts the higher will be the benefit. This means not just to reject possible solutions early, but to be aware of problems that might occur in later phases and be prepared for their solution. Often goal conflicts do not appear on abstract level, but due to decisions on a more concrete level. As the system is subdivided into many different subsystems of different domains and diffuse boundaries, it is difficult to trace relations without systematic assistance. Hence, the traceability is important to follow relations through a number of more or less concrete partial models.

In addition, the impact a certain change will have on the whole system can be estimated by following these traces. That means if a boundary condition changes during the development the analysis shows the effected areas of the product. It thus helps to decide on how and where to adapt the actual concepts or to start all over again.

The kind and number of relations of system elements can be analyzed. According to Daenzer[10] this gives helpful assistance in how to decide on system boundaries.

4. PARALLEL ROBOTIC SYSTEMS FOR HANDLING AND ASSEMBLY

Within the Collaborative Research Centre 562 “Robotic Systems for Handling and Assembly — High Dynamic Parallel Structures with Adaptronic Components” concepts for design and modeling of parallel robots for high operating speeds, accelerations and accuracy are developed. Due to the use of closed kinematic chains parallel robots feature relatively small moved masses (drives are mainly placed

in the rack) and high stiffness. In comparison with serial mechanisms they offer higher dynamics and high accuracy, especially when new and optimized structure components (e.g. adaptive joints[19] and rods[20]) are used. The disadvantages compared to serial robots are mainly a small ratio of workspace to installation area and the existence of singularities within the workspace. Thence, new design, analysis and control methods were developed to overcome theses drawbacks. As a mechatronic product several disciplines and many different partial models[21] are necessary to set the robot in operation. This results in relatively complex products with complex relations.

As by now few parallel robots are sold as mass products but customized to the needs of a special customer. The re-use of knowledge, thence the configuration through a modular concept and an effective change management through a systematic holistic view are helpful to provide the desired fast time-to-market as well as high quality and optimal products to the customer needs.

4.1. Product Surroundings

Figure 2(a) shows in the explorer view the packages of a model for the project "New Robot". Besides the SysML and UML profiles one can see the packages "Requirements" and "Surrounding". "Surrounding" documents the "product environments" and "Use Cases". The product environment contains elements the robot might interact with, e.g. conveyor belts or vision systems.

One use case in the product lifecycle phase "Use" describes the handling of muffins, which is one typical application for such a robot. Figure 2(b) shows two ABB IRB 340 Flexpicker[22] picking muffins of different type (light and dark) from a conveyor belt and placing two of each type into a box on another conveyor belt. The shown robots are of Delta structure, i.e. three identical kinematic chains with three rotational drives. They provide three translational degrees of freedom. In addition a fourth chain and drive provide a rotational degree of freedom around the vertical axis.

The last mentioned use case is not the only application for such a robot, thence all possible use cases are analyzed and its important parameters identified. The variety of these parameters influences the modules.

4.2. Requirements Structure and Relations

Figure 3 displays an excerpt of the concerning model elements and their relations. It shows that the use case "handling of muffins" leads to a refinement of the requirements "workspace" and "payload". A muffin weights up to 120 grams. Along with the gripper the required payload is around 1 kilogram. The workspace depends on the feeding, i.e. the area where to pick the muffin, the area where to place it and the required height to lift it (cp. Figure 2(b)). Geometry (cylindrical, cubic) and size of the workspace depend on the kind and arrangement of the kinematic chains. Thus, the block "kinematic chain" is able to satisfy the requirement "workspace".

Besides fulfilling the specific use case "Handling of muffins" one important goal is a short cycle time. There are a number of targets that support this goal. However, not every is related to the development

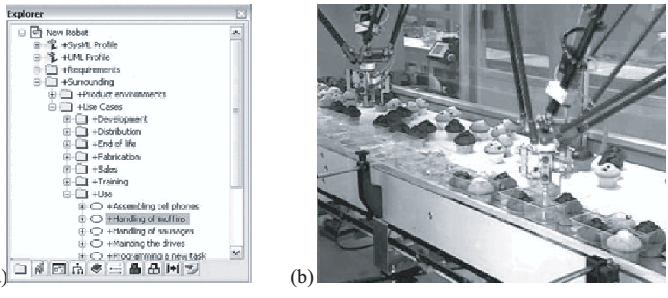

Figure 2. (a) Requirements and surroundings for a new robot in the hierarchical explorer view and (b) ABB IRB 340 in its production environment.[22]

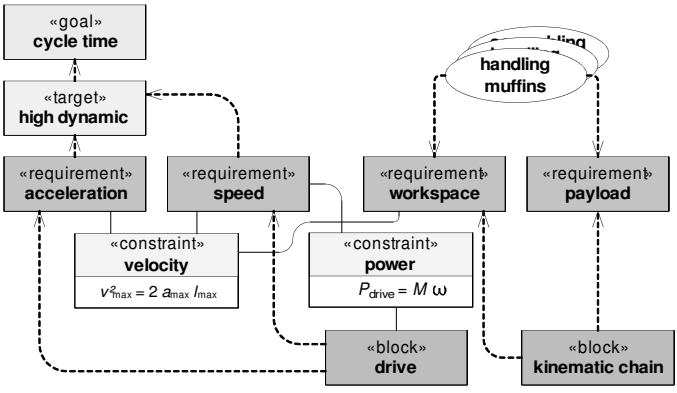

Figure 3. Goal oriented view on the product, excerpt of an extended requirements diagram based on SysML.

of the robot, i.e. not within the projects boundaries. The target "high dynamic" is directly related to the robot and supported by the requirements "high acceleration" and "high speed". As a simplified example at a low concretion level the triangular relationship between acceleration, speed, and workspace can be described by an equation considering constant acceleration at the tool centre point (TCP).

The requirement "high acceleration" can be satisfied by the drives. Balancing provided and needed power it is clear that the lower the moved mass is the higher gets the possible acceleration. But for a given payload (e.g. muffin and gripper) the kinematic chain has to provide a certain strength that results in a certain mass. Furthermore, a lightweight structure is more susceptible to oscillations that are excited by brake retardation, which might cause trouble for assembly operations with high needed accuracy. However, adaptronic components are able to suppress these oscillations.

Cost aspects are important to decide about the modularization strategy. Costs can be modeled in SysML in different ways. In a top-down approach they can be seen as cost- targets and integrated into the requirements model. In a bottom-up approach cost-estimates can be assigned to blocks as attributes. However, costs are influenced by a set of often dynamic and interfering boundary conditions. If the whole lifecycle should be considered (cp. TCO, LCC) a cost model has to be built up as a new partial model. Relations to the other partial models have to be drawn to analyze the costs. Cost modeling is subject to ongoing research and would go beyond the scope of this paper.

In later phases sometimes changes appear. For instance, a customer rethinks the cost target: The robot has to be cheaper otherwise he would back out. Then analyzing relations shows that a simple possibility is to exchange the drives by cheaper (but less powerful) ones. This would mainly affect the target "high dynamic" and to a certain degree the goal "cycle time". Depending on the specific substituting drive the target "low energy consumption" would be impacted, too. If the cheaper drive consumes as much energy as the more expensive one it would be up to the customer to decide, whether the cheaper robot justifies the loss in cycle time. If the cheaper drive consumes less energy this would provide an additional rationale considering the total cost of ownership (TCO).

4.3. Modular Systems for the Robot

The above considerations lead to a system of modular systems. Basic requirements like degree of freedom and workspace geometry lead to the choice of kinematic chains. The upper left part of Figure 4 shows a Delta structure with three identical kinematic chains for three translational degrees of freedom and an additional kinematic chain for the rotational degree of freedom. In contrast, a Hexa structure would use six identical kinematic chains for its six rotational drives.

The chain setup is important for the workspace characteristics. The right side of Figure 4 shows the chain setup of the Delta structure with rotational drive and universal joints. The universal joints allow a

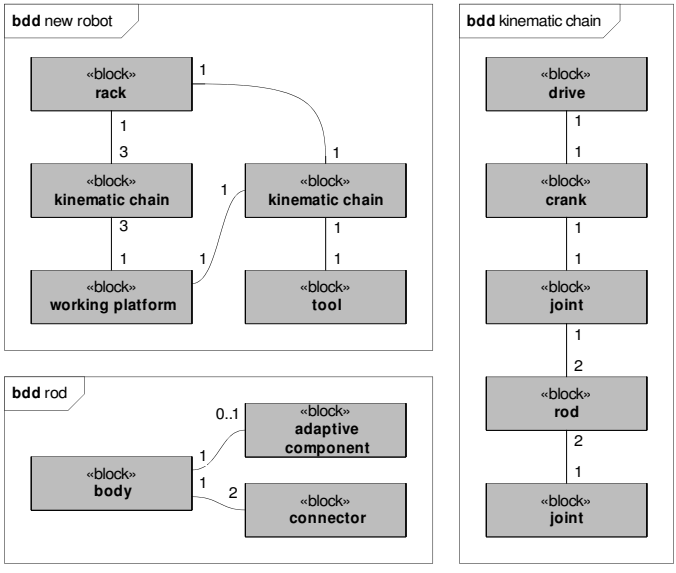

Figure 4. Examples of the modular structure in abstract SysML notation.

rod doubling to restrict the dof. A Hexa structure would use a universal and a spherical joint in its single-line chain. Thence, crank, joint, and rod are realized as modules of the kinematic chain. The lower left corner displays a rod that is made up of a body (a tube of certain diameter and length) and a connector at each side. For both, body and connectors, different variants are possible. Depending on the needed workspace the body length is adjusted. Depending on the orientation rod-joint (in and orthogonal to rod direction) different connectors are used (cp. Ref. 7). To support the target "active suppression of oscillations" adaptive components can be integrated on the rod body. However, this technology leads to additional costs in development, manufacturing, use and recycling. The consideration of the product surroundings show that the technology leads to big benefits in the area of precise handling or assembly, namely to gain better cycle times. For some other use cases the benefit would not justify the costs. Thus, modules with and without this technology are provided.

A low moment of inertia of the accelerated components is a supporting requirement of the target "high dynamics". Here the strategy of oversizing would lead to too heavy and inert components. For instance, a crank could be designed to connect to the joint on different distinct positions, thus being flexible in the effective kinematic crank length. For most applications the crank would be too heavy and the energy consumption of the whole system would be too high.

5. CONCLUSIONS

In nowadays market conditions many products became so called mass customization products, i.e. huge numbers of variants has to be generated while the total number of sold products is relatively small. After describing the most common strategies for developing modular systems a modeling approach using the Systems Modeling Language (SysML) is shown. The approach focuses on the generation of a requirements model as a basis for discussion and analysis of the real project aims. A short and simplified example from the field of parallel robots illustrates the approach.

ACKNOWLEDGMENTS

The authors gratefully thank the German research association (DFG) for supporting the Collaborative Research Centre SFB 562 "Robotic Systems for Handling and Assembly — High Dynamic Parallel Structures with Adaptronic Components".

REFERENCES

[1] Firchau, N. L. (2003). Variantenoptimierende Produktgestaltung, Ph.D. Thesis, Intitut für Konstruktionstechnik, Technische Universität Braunschweig.

[2] Franke, H.-J., Hesselbach, J., Huch, B. and Firchau, N. L. (2002). Variantenmanagement in der Einzel- und Kleinserienfertigung, Hanser Verlag.

[3] Pimmler, T. U. and Eppinger, S. D. (1994). Integration analysis of product decompositions, *ASME 6th International Conference on Design Theory and Methodology*, pp. 343–351.

[4] Franke, H.-J. (1976). Untersuchungen zur Algorithmisierbarkeit des Konstruktionsprozesses, Ph.D. Thesis, Institut für Konstruktionslehre, Maschinen- und Feinwerkelemente, Technische Universität Braunschweig.

[5] Pahl, G. and Beitz, W. (2007). *Engineering Design*, Springer Verlag.

[6] Blees, C. and Krause, D. (2008). On the Development of Modular Product Structures: A Differentiated Approach, *International Design Conference (DESIGN08)*, pp. 301–308.

[7] Stechert, C., Franke, H.-J. and Wrege, C. (2006). Task-Based Modular Configurations for Hybrid and Redundant Parallel Robots, *International IFAC Symposium on Robot Control (SYROCO06)*.

[8] Ericcson, A. and Erixon, G. (1999). Controlling Design Variants: Modular Product Platforms, *Society of Manufacturing Engineers*.

[9] Stake, R. (2000). On conceptual development of modular products, Ph.D. Thesis, The Royal Institute of Technology, Department of Production Engineering, Division of Assembly Systems, Stockholm.

[10] Daenzer, W. F. and Huber, F. (Ed). (2002). Systems Engineering — Methodik und Praxis, Verlag industrielle Organisation.

[11] Ort, A. (1998). Entwicklungsbegleitende Kalkulation mit Teilebibliotheken, Ph.D. Thesis, Institut für Maschinenwesen, Technische Universität Clausthal.

[12] Avgoustinov, N. (2007). Modeling in Mechanical Engineering and Mechatronics — Towards Autonomous Intelligent Software Models, Springer-Verlag.

[13] Salustri, F. A., Eng, N. L. and Weerasinghe, J. S. (2008). Visualizing Information in the Early Stages of Engineering Design, *Computer-Aided Design & Applications*, 5, 1–4.

[14] The Official OMG Systems Modelseling Language (SysML) site (2007). http://www.omgsysml.org/

[15] Anggreeni, I. and van der Voort, M. C. (2008). Classifying Scenarios in a Product Design Process: A study towards semi-automated scenario generation, *Cirp Design Conference — Design Synthesis*.

[16] Bischof, A. and Blessing, L. (2007). Design for Flexibility: Making Provisions for Requirement Changes, *International Conference on Engineering Design (ICED07)*.

[17] Stechert, C. and Franke, H.-J. (2008). Managing Requirements as the Core of Multi-Disciplinary Product Development, *Cirp Design Conference — Design Synthesis*.

[18] Jung, C. (2006). Anforderungsklärung in interdisziplinärer Entwicklungsumgebung, Ph.D. Thesis, Lehrstuhl für Produktentwicklung, Technische Universität München.

[19] Stechert, C., Pavlovic, N. and Franke, H.-J. (2007). Parallel Robots with Adaptronic Components — Design Through Different Knowledge Domains, IFToMM World Congress.

[20] Rose, M., Keimer, R., Breitbach, E. J. and Campanile, L. F. (2004). Parallel Robots with Adaptronic Components, *Intelligent Material Systems and Structures*, 15(9–10), 763–769.

[21] Stechert, C., Alexandrescu, I. and Franke, H.-J. (2007). Modeling of Inter-Model Relations for a Customer Oriented Development of Complex Products, *International Conference on Engineering Design (ICED07)*.

[22] The ABB group on the internet, IRB 340 product information, 14.07.2008: http://www.abb.de/product/seitp327/262ce7c337e2b552c12570c9003ff7e6.aspx