Production Engineering manuscript No.
(will be inserted by the editor)

Reducing the model generation effort for the virtual commissioning of control programs

Reimund Neugebauer · **Uwe Schob**

Received: date / Accepted: date

Abstract This paper deals with the field of developing and deploying highly automated manufacturing systems. Especially the stage of the control programming suffers from few applicable verification methods, as existing approaches still include drawbacks. The virtual commissioning, as one example, requires additional effort in creating simulation models to being usable.

This work presents a method, which reduces the effort of generating simulation models through an automated reuse and transformation of existing development documents. In a detailed case study, a software prototype implementing the method was executed to derive a simulation model of an existing set of information. The used sources, their transformation and the results are described to give a qualified impression of the method's possibilities.

Keywords Simulation · Virtual commissioning · Computer aided engineering · Machine modeling

1 Introduction

Modern manufacturing plants are characterized as being mass production systems. The required productivity can only be fulfilled by highly automated, fast operating systems. As the produced goods additionally increase in complexity, their assembly processes are getting more complex as well. The development of such automated manufacturing systems is a challenging process as it requires the interweaving of several disciplines and the cooperation of different departments. Coordination problems, like misunderstandings or different assumptions, commonly arise.

Despite the principles of mechatronics as an integrated development with early property assurance [1], the documents for the mechanical construction, the electrical diagrams as well as control programs are created mostly separated. Especially in the field of special purpose machines, a sequential development process prevails. Often, the aspired product and process quality can only be achieved during the final assembly and startup phase. Occurring problems tend to involve time-consuming changes, which result in higher costs or longer waiting times until start of production.

In particular the field of control program development suffers from the increased system complexity. As it incorporates the activation, coordination and monitoring of most of the machine functionality, it represents a core aspect of every automation system. Although the share of software controlled machine functionality steadily increases, as seen in Figure 1, the means of assuring the requirements remain mostly unchanged. Diagnosing and evaluating a control program without the controlled components is rarely possible. Even if they are present for testing, high process velocities tend to exceed the human capabilities of perception and interaction.

To solve the above mentioned problems, current approaches are analysed and evaluated in section 2. Despite obvious benefits are they seldomly put to practice due to existing drawbacks. By applying and adopting a software design method to the field of engineering production systems, an alternate solution will be presented in section 3. The method provides the means to drastically reduce the effort to create machine simulation models needed for a virtual commissioning. In addition, the method can be utilized without changes to existing development workflows. A software pro-

Reimund Neugebauer
Fraunhofer Institute for Machine Tools and Forming Technology IWU
Reichenhainer Straße 88, 09126 Chemnitz
Tel.: +49371-5397-1404
E-mail: reimund.neugebauer@iwu.fraunhofer.de

Uwe Schob
Tel.: +49371-5397-1105
E-mail: uwe.schob@iwu.fraunhofer.de

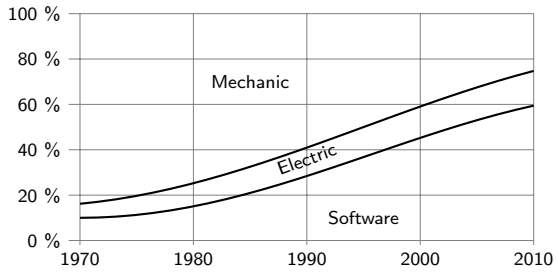

Fig. 1 Contribution of the different engineering disciplines to a machine's functionality according to [2]

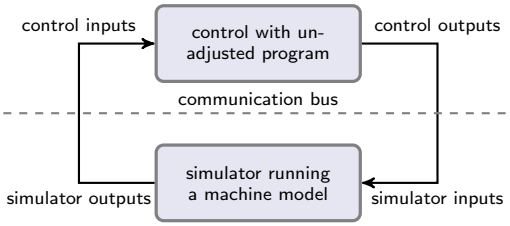

Fig. 2 A real control with an unadjusted program is connected to a simulated machine

totype implementing the method was used to perform a detailed case study in section 4, the results of which are then discussed in section 5.

2 Related work

The mentioned problems may be traced back to the development process itself. Therefore, a basic approach lies in the optimization of the development process towards concurrent and more abstract description paradigms. In [3], Bathelt extends the existing methods of the V-model-based design of [1] to bridge the gap between mechanical construction and control software. The mechanical construction in the form of the function structure is enriched by additional information modeling the information within a machine. These enriched documents can later be exported as a basic control program already containing most of the machine's sequences. This principle was also reused in [4]. Similar to that, Reinhart proposes a layered way of modeling systems [5]. Primitive elements available through libraries are grouped to generic mechatronic modules. Their composition to actual machine models is done in the more abstract layer of an engineering tool.

Remedying the inherent problems of the development process itself seems an appropriate way to deal with interdisciplinary coordination problems. However, a practical application in commercial available tools remains to be seen. The introduction of new development steps or the adjustment of established workflows might deem manufactures too chancy.

Another, more practical, approach to increase the quality of control programs is an early program test using a virtual machine model. This allows the simulation of unavailable machine components without changing the controlling program as seen in Figure 2. This technology is called virtual commissioning and widely known among manufacturers. It may be applied to different kinds of controls like numerical controls or programmable logic controls (PLC), see also [6]. Its benefits are well documented, as in [7]. Various software tools are established, that support defining virtual machines and their inner logic. These definitions may be achieved by different modeling means, like proprietary logical linked signal flows, modeling standards as *Modelica* [8] or even integration of compiled C-code. The main drawback of theese approaches is the additional effort of modeling the automation system as stressed in [9] and [10].

The authors conclude, that current approaches are not sufficient to meet the requirements of developing highly complex production systems with an assured quality in a minimal timespan. A newly approach has therefore to fulfill two main goals, not solved in conjunction so far:

- reducing the effort for creating simulation models for the virtual commissioning and
- being usable without change to existing, possible sequential development processes.

3 Approach

During a resarch project at the Fraunhofer Institute for Machine Tools and Forming Technology, the possibilities of resolving the shown issues were examined. As almost all information generated during the development process is available in a digitalized form, it can be accessed easily. Its reuse, instead of re-input, was the primary condition of this work. The information concerning a system development is separated into different documents of the different engineering disciplines. Analysing these documents can yield useful results concerning inconsistencies within an ongoing development, as described in [11]. In conjunction, all of these documents form a mechatronic view of the manufacturing system to be developed.

Taking this into consideration, one has to be reminded, that the generation of simulation models is based on such a mechatronic view as well. Albeit different in its sources, a simulation expert acquires information of the system to be modeled. After deciding about important aspects, he rebuilds relevant behaviour by the means of a simulator. This principle is shown in Figure 3, where a virtual machine is based on three different source documents.

Although the concept of models is common knowledge, formal definitions or specifications are rarely found in the field of machine simulation. Looking into the field of software technology, models itself are set as main focus. The

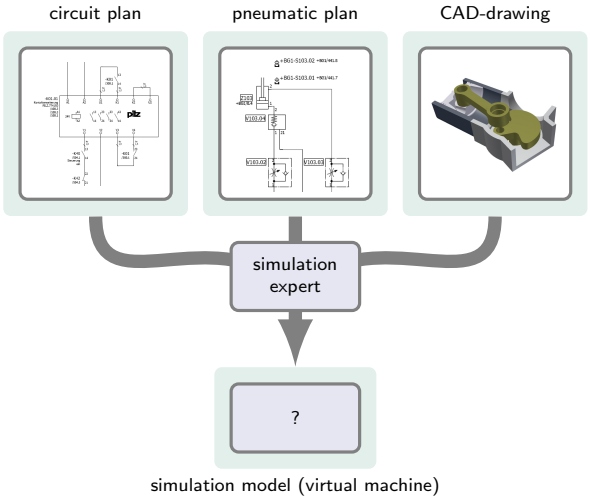

Fig. 3 A simulation expert performs a manual interpretation of different product documents to create a simulation model

Model Driven Architecture (MDA) is an approach to enable the consequent use of models in defining key aspects of software applications [12]. One important aspect of the MDA is the automatic refinement of abstract system definitions to platform specific descriptions. Without going into too many details, Figure 4 shows this principle, where a model of one domain is transformed into another domain through rules defined on the level of meta-models. Despite the fact, that its main intention is to formalise the application development, it may also be applied to other fields of research. For this work, the MDA was used as a guideline to formalise the process of generating simulation models.

Adopting the MDA to the field of generation of simulation models is the main part of the realised research project. The steps performed by a simulation expert are formalised and put into rule-based algorithms. They can be executed with nearly no manual interaction and lead to one or more simulation models. This is refered to as transformation process, as described in [13].

The transformation takes place when a sufficient amount of information of the system to be simulated is available. Possible sources of information are, but not limited to:

- the mechanical construction with geometric bodies and their relations,
- the electrical plans with components like sensors, switches and actuators and their wirings or
- the fluid plans with hydraulic or pneumatic components and their connections.

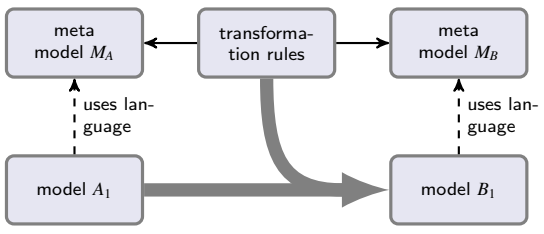

Fig. 4 Model-to-model transformation on meta model level [12]

The commonality of all these partial models is, that they contain objects with attributes and relations between them. Albeit the object's meaning differ from domain to domain, they all suffice in the context of the MDA. Transformation rules, that address sets of objects, may now be formulated. These rules represent the knowledge of the simulation expert to find specific patterns in the available information sources. Once found, a pattern is translated into one or more objects in the domain of a simulation tool. The repetition of this pattern search with all supplied rules then results in a simulation model. It has to be remarked, that its completeness depends solely on the completeness of the information sources. The existence of consistent data sources is therefore a requisite to a successful transformation.

To being able to verify this approach, a software prototype was implemented, see [14]. It represents an engineering tool, through which the information sources to use may be specified. Additionally, it allows the definition of transformation rules and different output models. Its immediate use is demonstrated in the following section.

4 Case study

In a first attempt to capture the possibilities of the method, a small part of a larger automation system was used. Its development documents were taken as the souce of information for the following transformation process. Figure 5 shows a CAD-drawing of a system, the task of which is to grab different product parts and move them to a type-specific position. The linear drive, the vertical stroke unit and the gripping module are arranged as an open kinematic chain. The gripping module is internally driven by a more complex closed kinematic chain.

The subtle reader may notice, that the figure does not show any transportation to and from the station. For the purpose of demonstration is this an intended inconsitency between the CAD-drawing and the electric plans, where appropriate handling devices are prepared. Further inaccuracies between the system parts, which are described by the documents, will be shown.

4.1 Data sources

As already mentioned, the source of information for a model transformation is a set of digital documents, created during the developmet process. Their content is analysed and rearranged in a simulation model. In order to being useable for

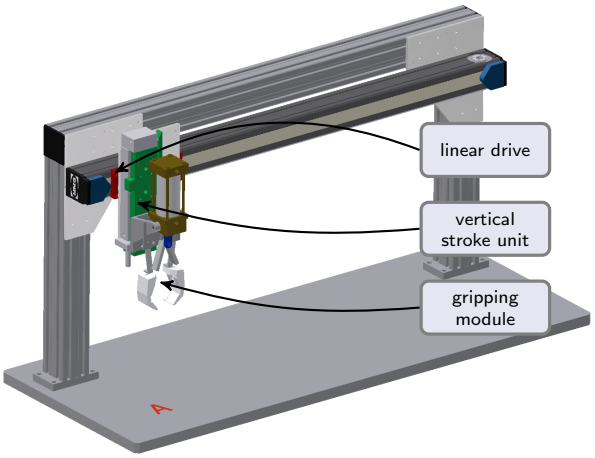

Fig. 5 Example of an automation system designed for the sorting of incoming product parts

the pupose of the virtual commissioning, the model needs to fulfill some conditions.

One important aspect to remember is, that the virtual machine needs to ”look” like the real counterpart for the control. As it operates through an image of inputs and outputs, these signals are to be simulated correctly. Their sequence is based on the interconnection of the machine's components, e.g. electric, pneumatic or physical, and thus needs to be modeled accordingly. For this exemplary transformation, the following machine aspects are used:

CAD-drawings contain objects that describe physical bodies. They have various attributes, like mass, center of gravity, density and inertia. Additionally they define surfaces, that are actually used to display them graphically. Relations between bodies are defined through constraints, which limit the possible movements between two components. If bodies are not fixed to each other, they contain one or more remaning degrees of freedom. Analysing all bodies and their relations can therewith result in the machine's underlying kinematic structure. Simulating this structure is necessary to being able to generate correct sensor signals which are dependant on positioning information of a mechanism's output-side.

Electric plans contain the largest part of a machine's inner logic. They comprise of standardised symbols, which are connected through wires. Additional relations exist, since symbols with identical reference identifications on different pages still belong together and form one physical component. The electric plans contain the information, which of the control's input/output signals is connected to which sensor/actuator. Therefore, these plans play a vital role in interconnecting the digital control to the mechanical aspect of a machine.

Fluid plans are very similar to electric plans, as they also describe how auxiliary power, in the form of pneumatic or hydraulic pressure, is utilized to perform mechanical work. The plans consist of symbols as well, except, that connections represent hoses instead of wires. As the software tool used for generating the plans is able to manage electric and fluid plans in one large document, further distinctions are not necessary.

Manual complements are necessary, to describe aspects of a machine, not documented elsewhere. For instance, what geometric part is driven by a pneumatic cylinder is only given in an informal notation through textutal annotations. Another example is the material flow within the system. Described only in textual form, no formal definition of it exists. It is therefore necessary to complete the existing information with manual ones.

4.2 Transformation

All of the above information is described as documents in individual software tools. By using the respective applications themself, the information can also be accessed algorithmically. For example, the CAD-drawings are read from the application *Autodesk Inventor 2009*, which comes with an own application programming interface (API).

A newly developed software tool acquires the objects, attributes and relations through APIs and stores them internally. In a second step, every of the predefined rules will be matched with the internal data. If a subset of the source objects match a rule, a new object, but now designated for the simulation model, will be created. Figure 6 shows, how a pneumatic valve can be found in a pneumatic plan. It consists of the 5 shown raw symbols, which needed to be covered by an appropriate rule. If the rule is matched, the element at the right side is created as part of the final simulation model.

As a third step in the transformation process, all formal connections in source documents are applied to the respective simulation elements. By this final step, a usable simulation model is created.

Along with the rules for the electric aspects, additional ones exist for analysing the kinematic system of a machine. Constraints that are applied to mechanical bodies can be aggregated and translated into one or more of the following joint types, see [15]:

- revolute joint,
- prismatic joint and
- spherical joint.

In result, the simulation model can contain the appropriate multi-body equivalent of the machine's kinematic. The

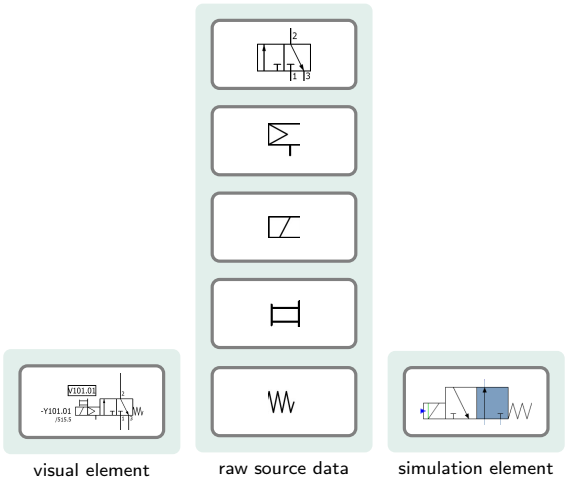

Fig. 6 Grouping of elements during transformation

vast majority of the mechanical parts are grouped during the analysis, as they are fixed to each other. This reduces the calculation effort, allowing even the simulation of dynamic properties based on forces.

It has to be remarked, that the transformation requires simulation libraries which can represent enough of the machine's components and their behaviour. Even though the concept and design of such libraries have a major influence on the success of a transformation, the current article assumes the existence of appropriate libraries. Their design is beyond the scope of this work.

The transformation process itself can be performed completely autonomous, as long as the input information are consistent. If the process encounters source document objects, that are not matched by any rule, appropriate feedback is given. This happens as well, if actuators in the elecric plan are not linked to any remaining degree of freedom between two geometric objects and vice versa. A handling device, which is not shown in the CAD-drawings, as described at the beginning of section 4, results in several warnings about unconnected mechanical outputs. After resolving the inconsistencies in the source documents the process can be restarted by a mere button click.

4.3 Resulting simulation model

The exemplary automation system consists of mainly two documents, a CAD assembly and a joint elecric and pneumatic plan. Table 1 and 2 give an overview about the amount of data available in the source documents.

The analsyis of the CAD-drawings yielded 26 part instances, which were grouped to 9 assembly units. These units consist of part instances which are fixed to each other and thus can be considered as one group. Between the groups remain various degrees of freedom, which represent the actual movable machine components. Figure 7 shows an effective constraint graph, where each group is represented as a node and the connections between them are general joints. The dashed lines indicate closed kinematic chains, which need to be treated differently regarding the resulting simulation model. For a better understading, Figure 8 shows a section of the assembly document, where each part instance is colored according to its group affiliation.

One section of the created simulation model contains the multi-body system, which represents the above mentioned groups as seen in Figure 9. The driven joints are depicted with short black lines at the bottom, where the connections to the driving elecric or pneumatic elements will be attached. The elements' parameters, like mass and inertia, are derived directly from the CAD-drawings and are used for calculat-

Table 1 Overview of the size of the used CAD-diagrams

type	count
different part-types	21
part instances	26
grouped assembly units (after analysis)	9
general joints between groups	11
thereof driven	3

Table 2 Overview of the size of the used circuit and pneumatic plans

type	count
document pages	87
thereof relevant for transformation	48
identified components	141
raw symbols	1313
connections (wires and tubes)	660

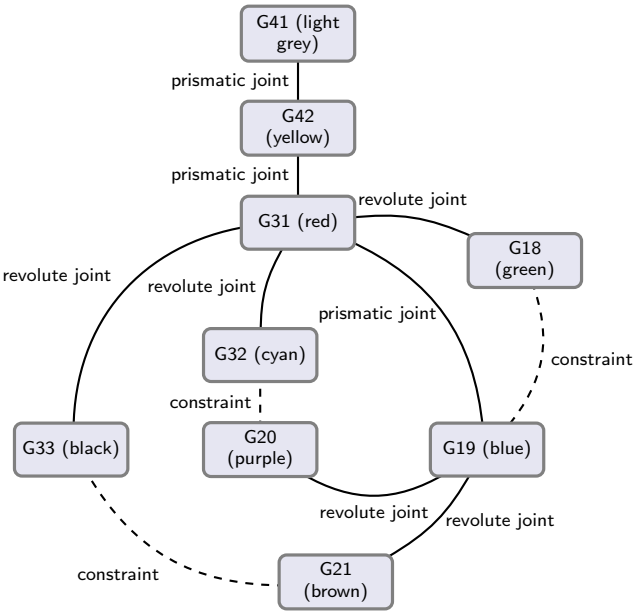

Fig. 7 Effective constraint graph of the demonstration graph

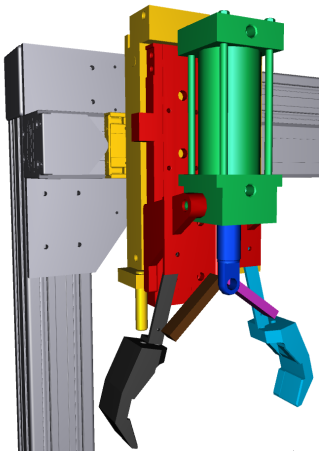

Fig. 8 Parts coloured according to their group affiliation

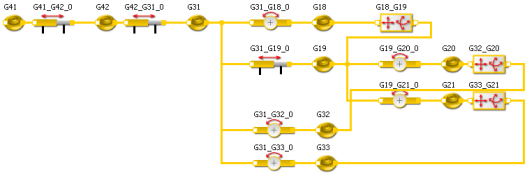

Fig. 9 Resulting simulation model of the kinematic system

ing the system's dynamic properties, e.g. position and orientation, [16].

As a side product of the CAD-drawings, a document containing the bodies geometric surfaces is created. It can be used within a 3D-viewer to enable a visual feedback of moving components during simulation.

Beside the mechanical aspect, the machine's inner logic plays an important role as well. To being able to cooperate with a real control, communication elements are required as part of the simulation model. These are created during the transformation process like any other elements. The appropriate rules look for PLC connection points within the electric plans. They represent the interface between control and hardware and will be substituted by an element, which operates by the means of OPC, see also [17]. The simulator bundles all communication elements and connects to the respective data object within the control itself. The addressing information is derived from the electric plans, where the correct byte or bit address should be present. Figure 10 shows a simplified model, which uses a control's output to operate a pneumatic cylinder. Its position is monitored by a sensor, which in turn signals it back to the control as an input.

However, the communication is only a smaller part of model. The major part consists of electrical switches, safety components and various actuators and sensors. Their interconnection forms the machine's switching logic. Figure 11 depicts the transformation result of a single page with 3 pneumatic actuators and their activation.

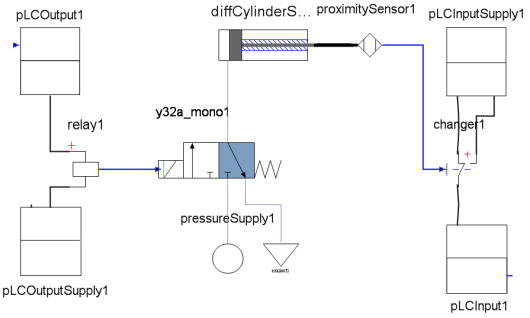

Fig. 10 Simplified simulation model with signal flow from the contol over electric and pneumatic elements back to the control

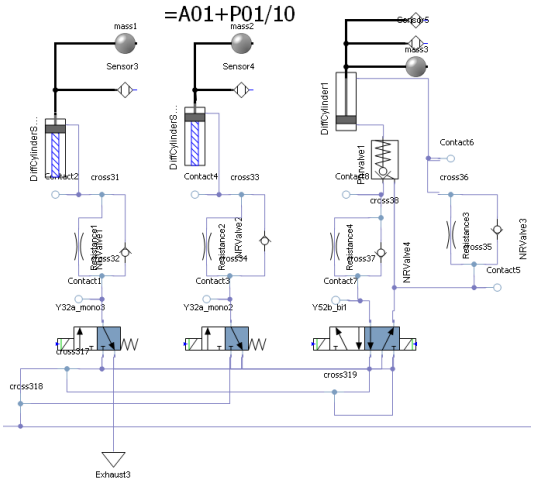

Fig. 11 Simulation model generated off a section of the pneumatic plan

Table 3 Resulting number of simulation elements

objects	count
raw symbols used	1009
connections applied	633
components created	117

In total, the transformation process was driven by 46 rules, which can be reused and extended for other systems. As seen in Table 3, the mapping of source objects to simulation elements is not 1 : 1. Some symbols are only used for graphical purposes, e.g. device cabinets, others have no purpose in a simulation, e.g. open sockets and terminals.

4.4 Performing the virtual commissioning

After the completion of the transformation process, the virtual commissioning can be performed. As the simulation model contains all necessary elements to communicate with the control hardware, it needs only to be started. The control can execute the desired program normally. The communication is handeled in the background, as shown in Figure 12.

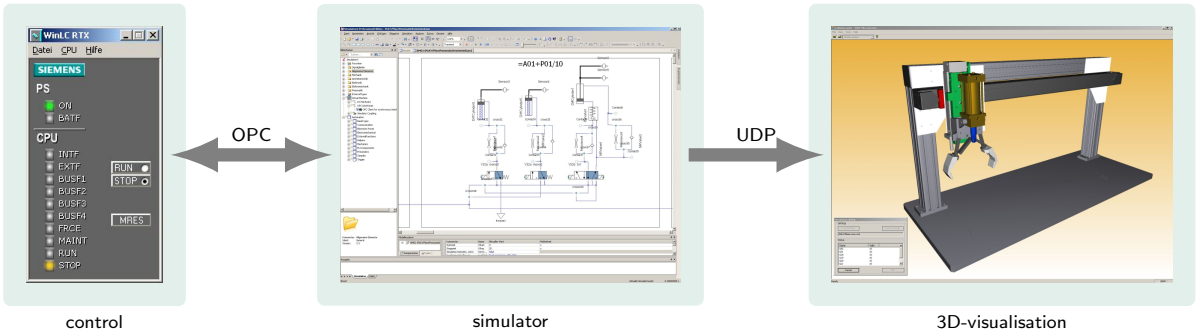

Fig. 12 Performing the virtual commissioning through the coupling of a control with a running simulation model and a 3D-visualisation

Additional data is transmitted to a 3D-visualisation to update positional information of moving parts.

Through the standard means of the control itself, an interaction with the simulated system is possible. This means, human machine interface (HMI) software operates as usual and allows the direct control of actuators. In addition, the simulator allows tracing and diganosing of signal curves. Some parameters can even be adjusted during the simulation.

During the simulation runs, several errors were identified. They are either to be found in the source documents or the control program. The following list shows an extract of the errors, that can be identified:

- Sensorsignals, which are used for continueing the program sequence, are eventually swapped. If the control program does not continue its expected sequence allthough its preconditions are met, it possibly uses the wrong input or output signal.
- The driving of actuators in the wrong direction, e.g. extending a cylinder instead of retracting, is another common problem. It can be seen in the 3D-visualisation and easily be corrected.
- If the control program uses correct signals, ongoing sequence errors may be traced back to electrical or pneumatical wiring problems. These occur if wires are connected to the wrong PLC input or output connectors.
- Parts, that are only insuffciently constrained to other objects of the CAD-drawings, may result in warnings during the creation of the multi-body system.
- Synchronized movements of different actuators can be visualised. In combination with collision detection algorithms, the geometric feasability of the parallel operation can be evaluated.

5 Results

The presented approach has shown, that it is possible to reuse development documents to derive simulation models thereof. With a small number of pattern based rules, the major part of a machine's functionality can be transformed automatically. As the approach is based on the original development documents, no conversion are necessary and real properties or attributes can be used.

Allthough manual input is needed in place for information not available in formal documents, an inclusion of other sources is intended. Even considering hydraulic plans or early specifications, like the extended function structure, see [3] or section 2, is possible. Similar to the source documents, can the output documents be adjusted. By using different libraries, the level of detail of the simulation models may be varied as well. This should be remembered, as the actual used libraries only partial fulfill hard realtime requirements. Current analysis suggest to restrict the model's size to separate sections. The usage of more high-grade simulation equipment could extend the limitations, but needs further evaluation.

The transformation can be executed at different stages of the development process, enabling an iterative improvement of the resulting simulation models. This is not achieved through manual changes of the results but instead by improving the underlying source of information, which increases their overall quality as well. The rules used for the presented automation system can be reused and extended for other machines.

6 Conclusion

This paper has presented a general approach for automating the generation of machine simulation models. The approach is based on a model-to-model transformation on a meta model level. The source of information are documents created during the normal development process. The results of the transformation process are simulation models representing different aspects of a machine.

Extendable sets of rules define, how source elements are transformed into target elements. Although the approach aims to be fully automated, the level of automation depends on the available sources. Information available in a formal

way can be used without restrictions. Unclear or undocumented information, like associations between an actuator and its driven mechanical part, require manual user input.

By this means, the effort in creating virtual machine models can be reduced significantly. The virtural commissioning can be performed at an early stage. Design flaws and software errors can be uncovered with less effort, thus increasing the overall software quality before the system's startup.

As the main transformation process includes the acquisition of various data, further work should focus on the integration of existing mechatronic models like *AutomationML* [18]. The automatic population of such interdisciplinary formats would allow a manufacturer to use his existing knowledge in conjunction with emerging technologies as well.

References

1. "Vdi 2206 - entwicklungsmethodik für mechatronische systeme," Verein Deutscher Ingenieure, Jun. 2004.
2. J. Glas, *Standardisierter Aufbau anwendungsspezifischer Zellenrechnersoftware*. Springer Verlag, 1993.
3. J. Bathelt, "Entwicklungsmethodik für sps-gesteuerte mechatronische systeme," Ph.D. dissertation, Eidgenössische Technische Hochschule Zürich, 2006.
4. J. Botaschanjan, T. Hensel, B. Hummel, A. Lindworsky, M. F. Zäh, G. Reinhart, and M. Broy, "Autovibn - abschlussbericht: Automatische generierung von verhaltensmodellen aus cad-daten für die qualitätsorientierte virtuelle inbetriebnahme," Technischen Universität München, Institut für Informatik, Tech. Rep., Jun. 2010. [Online]. Available: http://www4.in.tum.de/~hummelb/publ/autovibn-2010.pdf
5. G. Reinhart, T. Hensel, A. Lindworsky, and M. Spitzweg, "Teilautomatisierter aufbau von simulationsmodellen," *wt Werkstattstechnik*, vol. 97, pp. 663–667, 2007.
6. R. Neugebauer, D. Weidlich, K. Wegener, and A. Kunz, Eds., *VR/AR-Technologien für die Produktion*, ser. Internationale Seminarreihe für industrielle Anwendungen von Technologien der virtuellen und erweiterten Realität zur Effizienzsteigerung in den Unternehmen. Technische Universität Chemnitz, Fraunhofer-Institut für Werkzeugmaschinen und Umformtechnik, May 2008.
7. M. F. Zäh, G. Wünsch, T. Hensel, and A. Lindworsky, "Feldstudie - virtuelle inbetriebnahme," *wt Werkstattstechnik*, vol. 96, pp. 767–771, 2006.
8. "Modelica - a unified object-oriented language for physical systems modeling," online, Modelica Association, Sep. 2007. [Online]. Available: http://www.modelica.org/documents/ModelicaSpec30.pdf
9. M. Bergert and C. Diedrich, "Durchgängige verhaltensmodellierung von betriebsmitteln zur erzeugung digitaler simulationsmodelle von fertigungssystemen," *Automatisierungstechnische Praxis*, vol. 7, pp. 61–66, Jul. 2008.
10. R. Drath, P. Weber, and N. Mauser, "Virtuelle inbetriebnahme - ein evolutionäres konzept für die praktische einführung," in *Automation*, 2008.
11. S. Altmann, U. Schob, and M. Winter, "Reale steuerung trifft auf simulierte anlage - datenkonsistenter design-prozess zur virtuellen inbetriebnahme mechatronischer systeme," *A&D-Kompendium*, vol. 2008/2009, pp. 67–69, 2008.
12. J. Miller and J. Mukerji, "Mda guide version 1.0.1," online, Jun. 2003. [Online]. Available: http://www.omg.org/cgi-bin/doc?omg/03-06-01.pdf
13. U. Schob, R. Böttcher, T. Blochwitz, O. Oelsner, and M. Winter, "Model based virtual startup of automation systems," in *Proceedings of the 7th International Modelica Conference*, 2009.
14. U. Schob, "A framework for automating the generation of machine simulation models," in *World Academy of Science: International Conference on Control and Automation*, Nov. 2010.
15. J. U. Turner, S. Subramaniam, and S. Gupta, "Constraint representation and reduction in assembly modeling and analysis," *IEEE Transactions On Robotics And Automation*, vol. 8, pp. 741–750, 1992.
16. T. D. Krupp, "Symbolische gleichungen für mehrkörpersysteme mit kinematischen schleifen," Ph.D. dissertation, Gerhard-Mercator-Universität, 1998.
17. F. Iwanitz and J. Lange, *OPC - Fundamentals, Implementation and Application*. Hüthig Verlag, 2006.
18. "Automationml specification part 1 - architecture and general requirements," online, AutomationML e. V. c/o IAF, Jan. 2009. [Online]. Available: http://www.automationml.org/forum/download/file.php?id=28