

# A COMPONENT ORIENTED APPROACH TO MULTIDISCIPLINARY SIMULATION OF MECHATRONIC SYSTEMS

**Roland Kasper<sup>1</sup>, Dmitri Vlasenko<sup>1</sup>, Gennadi Sintotskiy<sup>1</sup>**

<sup>1</sup>University of Magdeburg, Faculty of Mechanical Engineering,  
39106 Magdeburg, Universitätsplatz 2, Germany

*Roland.Kasper@mb.uni-magdeburg.de (Roland Kasper)*

## **Abstract**

Due to the multidisciplinary and complex structure of mechatronic products, their complete modeling and simulation is a hard challenge. On the other hand complete models are a key element of the development process of mechatronic systems. Starting from proofed modeling methods for mechanical, electrical and software systems, an integrated component oriented method for the modeling and simulation of multidisciplinary mechatronic systems will be presented in this paper. The basic concept uses components that represent shape/geometry and functions of real world components. Well defined external interfaces allow combining several components to build up more complex ones leading to complete systems in a straightforward bottom up process. The automated generation of multidisciplinary simulation models in semi-symbolic form from extended 3D-CAD models is a key element of the method that guarantees the consistency of generated models avoids errors of manual modeling and reduces modeling effort significantly. Applying the generation process on single components, result in very compact and efficient simulation code. The models are used on a mixed continuous / discrete simulation platform that preserves the component structure of the models during simulation, which allows the simulation of large systems on the basis of verified precompiled sub-components. The support of modules, usually restricted to the specification of mechatronic systems, in our approach is continued to the simulation model which results in encapsulated robust simulation models.

**Keywords: Multidisciplinary modeling, component orientation, mechatronic system modeling.**

## **Presenting Author's biography**

Roland Kasper got a Diploma of Mechanical Engineering from the Technical University of Karlsruhe in 1978. Subsequent to a Dr.-Ing. Degree from the University of Paderborn in 1986 he was working with Robert Bosch Company as a researcher and senior researcher for eight years. Since 1994 he is a full professor of Mechatronics at the University of Magdeburg, Faculty of Mechanical Engineering. Further he is a member of the Saxonian Academy of Science at Leipzig, a member of the grant committee for collaborative research centres of the DFG (German Research Foundation) and a member of the acatec.



**Acknowledgement:** This project is funded by the DFG (German Research Foundation).

# 1 Introduction

The past years showed a large number of very ambitious and successful mechatronic products e.g. in the field of automotive systems, in robotics or in medical technologies. The intelligent integration of mechanical, electrical and software functions into one product is an important precondition for the realization of new unachieved product features at a competitive level of costs and quality. Ongoing improvements, first of all in the area of micro-electronics but also in the area of integration technologies are a strong indicator that this successful fusion of mechanics, electrics and software will remain a permanent booster of future development.

However, development and test of mechatronic systems creates new challenges for design and test engineers, because traditional design methods, well known and proofed in mechanical or electrical engineering, cannot fulfill the demands given by multidisciplinary mechatronic products. An integrated simulation support of the design and test processes of mechatronic products has shown to be one of the key elements to overcome these problems [1]. Integration in this context addresses two aspects. First, integration of functional models across different disciplines to allow an overall simulation of systems built up from mechanical, electrical and software parts. Second integration of shape and geometry oriented 3D-models with corresponding functional models. Both aspects are important because only a functional and geometric optimized product will be competitive. Today, 3D-CAD tools support the specification of the geometry of mechanical parts together with the geometry and e.g. netlists of electrical parts like cable harnesses or electrical boards.

In this paper a concept is presented to utilize these specifications to generate consistent multidisciplinary simulation models in an automated step and use them on a continuous / discrete simulation platform to design and test mechatronic products. Geometry and material data of the 3D-CAD model are the source for simulation models of mechanical parts. In addition to their geometry and material data, electrical parts can be specified by netlists (e.g. SPICE netlists) that can be imported into the 3D-CAD model together with board and part geometry. Parameters of electrical components and electrical circuits are the source for simulation models of electrical parts. To support the integrated use of geometry and simulation models during the design and test of products efficiently, both models are coupled and animated in real-time. These coupled models represent geometry as well as function of a component or product with all interrelations.

A key element of the multidisciplinary modeling and simulation concept, presented in this paper, is the

strict component orientation. This allows simulating the design and assembly process of a real product, which can be built up from parts in a bottom up process, using existing components. On each intermediate level of the hierarchy all features and functions of real components can be realized and tested by simulating components. This is achieved using well defined interfaces between components like mechanical joints or electrical connectors that support an easy assembly process as well as a well structured way to generate equations to formulate the behavior of each component.

## 2 Methods and tools for modeling and simulation of mechatronic systems

### 2.1 Domain specific methods and tools

Because mechatronic products are dominated by the interaction of several physical domains, there is available a broad range of domain specific as well as of multi-domain oriented modeling and simulation tools.

For pure mechanical parts CAD tools come into action to specify geometry and material data. If 3D-models are used, these can be the source for the simulation of kinematics or multibody dynamics. However, the automatic generation of multibody models from a CAD model in most cases demands an expensive preparation of the CAD model to satisfy the preconditions of the generating tool with respect to allowed joints and constraints. Very often a completely new construction of a model suitable for simulation is necessary [2].

Also for pure electrical parts, there is a large number of powerful CAE tools, which generate standardized simulation models from a circuit diagram. A proofed and long used platform for the simulation, analysis and optimization of electrical circuits is offered by the SPICE standard [3], which is supported by nearly all tools available today. Newer standards like VHDL-AMS eliminate the restriction on pure electrical systems inherent to SPICE and allow the modeling and simulation of non-electrical parts. But the stability and efficiency of available simulators is still poor in the case of realistic mechatronic systems which include complex mechanical parts like big multibody systems.

### 2.2 System level

Due to the existing separation into mechanical and electrical parts on the level of CAD tools and the supporting tools for modeling and simulation, in many cases a signal flow oriented level is chosen for the simulation of the complete system. System design tools like Matlab/Simulink and many others which originate in the area of control systems, support the integration of mechanical, electrical and software parts into one system. The greatest drawback of this approach is the restriction to non-reactive signals and

a very limited support of reactive physical connections implemented by mechanical joints or electrical connectors. As a consequence, complete mechanical or electrical subsystems have to be modeled isolated in one domain before they are integrated into the signal flow of the complete system. This means an expensive effort and a source of modeling errors. While CAD models play an important part as an interface between development and production, signal flow oriented models are preferred to simulate overall functions of the complete system. Despite these restrictions, models on signal flow level are widely spread, particularly as a result of the insufficient integration of mechanical and electrical CAD into simulation tools.

In addition there exist a large number of tool-coupling and co-simulation solutions e.g. between Matlab/Simulink and mechanical simulators like ADAMS or electrical simulators like Mentor. But these solutions always only offer a coupling of physical models (mechanical or electrical) to signal flow models. Consequently the accuracy of these approaches is limited. Stability and efficiency of simulator coupling is still an unsolved problem.

Software development for mechatronic systems is supported by established microcontroller programming tools that cover the range from assembly to high level language as well as by newer graphical programming tools that are able to generate microcontroller code directly from a graphical specification of data and control flow. The integration of this code into the model of a mechatronic system today preferably is done on signal flow level as far as system development is concerned. Tools for mechanical systems in most cases only support to import behavioral models e.g. of a controller. Mixed mode simulators of electrical systems like MULTISIM allow the simulation of software together with a model of the used processor, which guarantees a very high timing accuracy but needs a huge amount of simulation time.

### 2.3 Multidisciplinary tools and methods

There are only a restricted number of tools dedicated to the modeling and simulation of multidisciplinary systems like ALASKA [4], which offers a strong modeling concept based on generalized Lagrange functions. Despite its strength for pure electromechanical systems this approach is not well suited to model and simulate electrical circuits, with a large number of switching elements and large differences in time constants.

A very general approach to multidisciplinary modeling is offered by symbol manipulating tools like DYMOLA [5]. Its object-oriented modeling language MODELICA offers a structured and easy to use multidisciplinary modeling platform. The symbolic generation of system equations allows their use in very different environments. However, symbolic

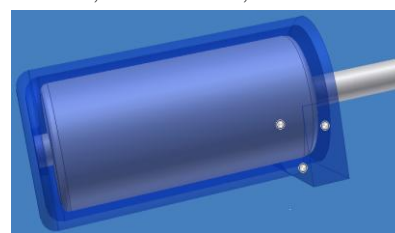
equation generation has its limits, as the number of equations and variables explodes for larger models.

### 2.4 Subsystems and components

All common methods and tools used today support some kind of subsystem hierarchy or even a hierarchical object model on specification level. This is absolutely necessary to deal with large and complex systems during specification. But during the generation process of the numeric or symbolic simulation model the hierarchy and substructures of the specification model is flattened. The final simulation model consists of one level of variables, parameters, equations, matrices etc. which has to be generated by scratch even if only a small sub-component has changed. Thus reuse of subsystems is restricted to the specification level. Well known and proofed elements of software design like precompiled components (e.g. procedures or classes) cannot be used for simulation and their advantages like reduction of verification effort or the building of

#### Geometry Model

- Mechanical Properties
- Geometry and Material
- Mass, Center of Mass, Mass Moment of Inertia



#### Interfaces

Motor Shaft

Connectors

Motor Supports

+ Electrical properties

+ Resistance, Inductance, ...

+ Cable Harness, Connectors, ...

#### Electrical Model

Fig. 1 Extended 3D-CAD model of a DC-motor

verified libraries of simulation models cannot be exploited. This is the reason why components play an important role in the concept of multidisciplinary modeling and simulation presented in this paper.

## 3 Component Oriented modeling

### 3.1 Multidisciplinary specification

Recent advances in CAD tools make the separation between different domains disappear step by step. The latest versions of tools like Autodesk INVENTOR or ProE not only support the complete specification of geometry and material data of mechanical parts or assemblies, they also offer the possibility to specify electrical parts, cable harnesses and boards. Mechanical and electrical features and parameters of electromechanical components thus can be specified together in one 3D-CAD model.

Fig. 1 shows an example of a DC-motor which includes geometric and electric model data.

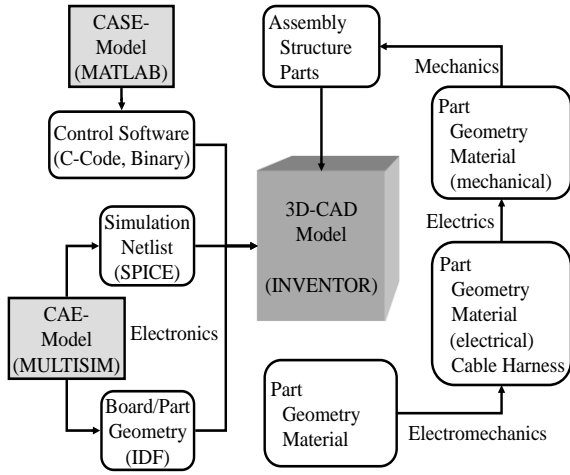


Fig. 2: Extended multidisciplinary 3D-CAD model

Additionally electro-mechanical interactions like the torque produced by the magnetic

field and the motor current or the induced voltage generated by the magnetic field and the angular velocity of the rotor consequently can be attached to the air gap of the motor which is an explicit part of the 3D-CAD model. To be used as a component real mechanical interfaces like the coupling of the motor shaft or the mounting of the motor housing as well as electrical interfaces like the connector can be specified together with the geometry.

Integration of electrical components, circuits and boards, originally specified inside electrical CAE tools, can be accomplished by importing 3D-part and board data into a mechanical 3D-model. With this extension modern 3D-CAD tools offer a possibility for the integrated specification of mechatronic parts and assemblies, collecting all geometry, material and electrical data of a part in one extended multidisciplinary 3D-model as shown in Fig. . Furthermore these tools allow an easy mounting of parts and assemblies to more complex assemblies with a hierarchical bottom up approach.

In the same way, control software for a microcontroller can be imported as a high level language or as binary file and attached to a 3D-CAD model of the microcontroller. Ports and pins of this model build the interface between the software and typical electrical parts like wires, transistors or complete circuits specified in the electrical domain. In this way the extended 3D-CAD model can be complemented with software elements.

From this point of view extended 3D-CAD models offer an ideal level for the specification of mechatronic systems. On the other hand CAD models do not represent the function or behavior of a part and therefore they cannot be used for simulation directly. From a simulation point of view the question of generating simulation models for these components is of highest importance.

### 3.2 Mechanical parts and components

Each mechanical component can be seen as a system consisting of multiple rigid bodies. The coordinates of all internal bodies that have no external constraint are arranged in the vector  $q_i$ , while the coordinates of all bordering bodies with at least one external constraint are arranged in the vector  $q_b$ . Following Newton-Euler formalism the accelerations of the bordering bodies generally can be written as

$$\ddot{q}_b = D \cdot f_b + d, \quad (1)$$

where  $f_b$  are the reactive forces associated with the external constraints of the component. The elements of the matrix  $D$  only depend on the coordinates and the velocities of the component. Additionally the vector  $d$  also includes external forces. Eq. (1) is referred to as the semi symbolic interface of a mechanical component.

Building up this parent component from several sub-components (children), represent the bottom up formulation of the concept of mechanical components. Fig. shows an example of this situation with a parent component containing two bordering components and three internal sub-components. The accelerations of all internal components are arranged into the vector  $\ddot{q}_i$ , all internal reactive forces into the vector  $f_i$ . Taking into account the structure of the accelerations of the sub-components, vector  $\ddot{q}_i$  can be written as

$$\ddot{q}_i = D_{ii} \cdot f_i + d_i. \quad (2)$$

The accelerations of all bordering components are arranged into the vector  $\ddot{q}_b$ , all external reactive forces into the vector  $f_b$ . Generally internal as well as external reactive forces will act on bordering components. Thus the vector  $\ddot{q}_b$  can be written as

$$\ddot{q}_b = D_b \cdot f_b + D_{bi} \cdot f_i + d_b \quad (3)$$

Expressing the equations of internal constraints on the level of accelerations

$$G_i \cdot \ddot{q}_i + G_b \cdot \ddot{q}_b + g = 0, \quad (4)$$

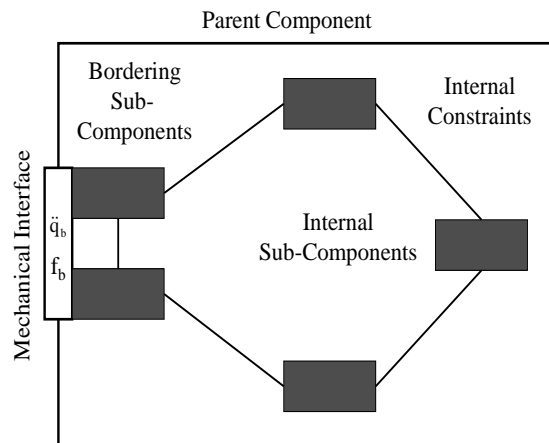


Fig. 3 Mechanical component built up from sub-components

eqs. (2)-(4) define a linear system of equations, which can be solved in a first step for the accelerations  $\ddot{q}_b$  in the desired structure of eq. (1). To finish the calculation of all internal accelerations of a mechanical component the values of the external forces must be available. These values are calculated by the parent components and then given back to the children. Usually a mechatronic product consists of large number of mechanical components that build a hierarchy from a number of basic components at the bottom up to one top component. The step defined by eqs. (2)-(4) can be repeated for each component consisting itself of sub-components. Basic components like a rigid body have predefined interfaces which can be defined using e.g. Newton-Euler formalism. Coordinates and velocities are the state variables of a mechanical component that have to be numerically integrated during simulation from the accelerations. As a consequence of eq. (4) which uses a formulation of constraints on the level of accelerations, a stabilization procedure is used during simulation [6]. A more detailed description of the algorithm can be found in [7] and [8]. Thus a mechanical component only has to know the semi symbolic form of the accelerations of its direct sub-components and receives its own external reactive forces from its parent. Consequently its equations only depend on the symbolic structure the interfaces of its children. In this sense a mechanical component is strictly encapsulated and can be implemented as a precompiled class and used by its parent component.

### 3.3 Electrical parts and components

For electrical systems the approach is quite similar. Each component can be written as a multipole which has a number of external pins whose voltages can be arranged in a vector

$$U_p = U_0 + R \cdot I_p + L \cdot \dot{I}_p, \quad (5)$$

where  $U_0$  is the vector of internal absolute voltages generated by voltage sources or capacitors and  $I_p$  is the vector of pin currents.  $R$  and  $L$  are matrices containing resistances and inductances of the pins respectively. In general the matrices  $R$  and  $L$  are not constant but depend on specific currents of the component. In this case eq. (5) is nonlinear, but nevertheless is valid. For most mechatronic applications matrix  $R$  can be treated at least as piecewise constant matrix whose elements change dependent on the state of one or several switching elements like diodes or transistors. In this case it is important that the simulation method recognizes changes of state of these elements. The matrix  $L$  can be treated as known as its values will only depend on currents of inductive elements that are known state variables. In these cases eq. (5) is linear which will be assumed for the next steps. Eq. (5) is referred to as the semi symbolic interface of an electrical component.

Building up this parent component from a circuit

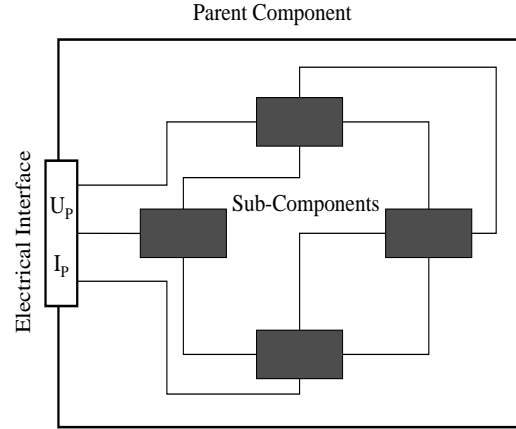


Fig. 2 Electrical component built up from sub-components

consisting itself of several sub-components (children) represent the bottom up formulation of the concept of electrical components. Fig. 4 shows an example of this situation with one connector building the electrical interface and 4 sub-components. The pin voltages of all internal components are arranged into the vector  $U$ , all pin currents into the vector  $I$ . Taking into account the structure of all pin voltages of the components, vector  $U$  can be written as

$$U = \hat{U}_0 + \hat{R} \cdot I + \hat{L} \cdot \dot{I}, \quad (6)$$

where the matrices  $\hat{U}_0$ ,  $\hat{R}$  and  $\hat{L}$  are defined block diagonal matrices of the corresponding matrices of the sub-circuits. Kirchhoff's laws applied to the internal part of the network deliver

$$P_U \cdot U = 0 \quad \text{and} \quad P_I \cdot I = 0 \quad (7)$$

with the incidence matrices  $P_U$  and  $P_I$ . For the external connectors of the component Kirchhoff's law can be written as

$$P_{PU} \cdot U = U_p \quad \text{and} \quad P_{PI} \cdot I = I_p, \quad (8)$$

with the incidence matrices  $P_{PU}$  and  $P_{PI}$ . Following the preconditions for linearity discussed above eqs. (6)-(8) define a linear system of equations. For typical component size in the field of mechatronic systems, the pin voltages of the component can be calculated in the structure defined by eq. (5) in symbolic form using MAPLE. To finish the calculation of all internal voltages and currents of an electrical component, the values of the pin voltages and currents of the component must be available. These values are calculated by the parent component and then given back to the children. Usually a mechatronic product consists of large number of electrical components that build a hierarchy from a number of basic components at the bottom up to one top component. The step defined by eqs. (6)-(8) can be repeated for each component consisting itself of sub-components. Basic components like resistors, diodes, transistors, capacitors and inductors have predefined interfaces. Voltages and currents are the state variables of an

electrical component that have to be numerically integrated during simulation from its derivatives. A more detailed description of the algorithm can be found in [9] and [10].

Thus an electrical component only has to know the semi symbolic form of the pin voltages of its direct sub-components and receives its own pin voltages and pin currents from its parent. Consequently its equations only depend on the symbolic structure of the interfaces of its children. In this sense an electrical component is strictly encapsulated and can be implemented and used as a precompiled class by its parent component. In the case of nonlinear elements in eq. (6) this symbolic solution can be used to feed an iterative solver for the nonlinear problem.

Additionally electric components provide an interface to control and information processing components e.g. via AD- or DA-converters, comparators, switches or muxers. The digital connection lines of these conversion and control elements are offered as event driven data lines and handled by the time discrete part of the simulator, which grants that analog values are sampled at exact time points efficiently and digital values impact analog simulation correctly [11].

### 3.4 Electromechanical parts and components

Following the approach presented for mechanical and electrical systems this can be combined very easily to implement electromechanical components. Electromechanical interfaces are a union of the presented mechanical and electrical interfaces. The program code behind an electromechanical component consists primarily of the code generated for the mechanical and the electrical parts. In addition interactions between the mechanical and the electrical domain are taken into account. Generally these interactions have a specific structure.

On the mechanical parts can act forces  $F_m$  and / or torques  $M_m$  that are generated by magnetic fields e.g. in electric motors or magnetic coil systems

$$F_m = F_m(\Phi, I, q, \dot{q}) \quad \text{and} \quad M_m = M_m(\Phi, I, q, \dot{q}), \quad (9)$$

with the magnetic flow  $\Phi$ , the actuating electrical current  $I$ , the coordinates  $q$  and the velocities  $\dot{q}$ . Similarly electric fields, e.g. in electrostatic or piezoelectric actuators, will act with forces and / or torques

$$F_e = F_e(Q, U, q, \dot{q}) \quad \text{and} \quad M_e = M_e(Q, U, q, \dot{q}) \quad (10)$$

with the electrical charge  $Q$  and the actuating voltage  $U$ . Both types can be modeled as external forces acting on the mechanical part of a component. Vice versa in electric parts will be induced voltages  $U_i$  and driven currents  $I_i$  generated by the movement of mechanical parts

$$U_i = U_i(Q, U, q, \dot{q}) \quad \text{and} \quad I_i = I_i(Q, U, q, \dot{q}). \quad (11)$$

Both can be modeled as voltage and current sources acting in the electrical circuit of a component. As it is

easily seen, coupling eqs. (9)-(11) depend only on the state variables of the mechanical and the electrical parts. Thus they can be provided in each integration step before accelerations or derivatives of voltages and currents are calculated.

This method to describe electromechanical interactions works for actuators as well as for sensors and delivers very accurate models. However in many applications electrical sensors for mechanical quantities allow a simpler approach, where the dynamic behavior of a sensor is approximated only by a behavioral model, e.g. a static sensor characteristic and a low-pass filter. Finally electromechanical components can use the interfaces provided by electrical sub-components to connect to control and information processing systems.

### 3.5 Digital control and software components

Specification of control and software parts of mechatronic systems as well as the generation of high level C-code for simulation or binary code for microcontroller implementation can be done by a number of tools, e.g. Matlab Real-time Workshop. These models are formulated in a way to read data from information processing interfaces at specified times, do their internal calculations either cyclic or interrupt / event driven and then write the results back to the interfaces at specified times. Usually models on simulation level only use simplified approximations of real sampling and output times with limited accuracy. As an advantage they are only a small burden for discrete event simulators. Models on the level of binary code, which are run using a simulator for the microcontroller itself, offer very high timing accuracy but need a very high number of discrete events to be handled by the simulator. Unattached of both approaches to the simulation of control and software systems, the resulting models can be encapsulated in classes. Because they only interact with interfaces of electrical and electromechanical components, control and software components are strictly encapsulated and can be implemented and used as a precompiled class by its parent component.

### 3.6 Component generation

In this way multidisciplinary components are the core elements of mechatronic models. They reflect the structures and sub-structures of the complete product across all levels of hierarchy. They offer well defined interfaces to assemble a product from parts and sub-assemblies and to simulate the model. Finally they build the link between the geometric CAD model and functions used during simulation. To guarantee consistency between the 3D-model and the component classes, all equations and simulation code will be generated automatically from the information presented by the extended 3D-CAD model. This is done straightforward for pure mechanical and electrical components following the methods described above. As shown in Fig. 5 information

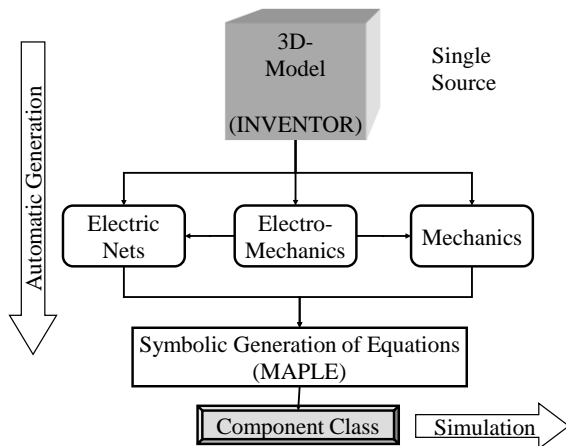


Fig. 4 Automatic generation of component classes

about electrical and mechanical elements are first separated and then treated in the usual way. The procedure used to generate equations takes care that all partial equations of a multidisciplinary component are composed in a correct way. Generating symbolic equations only for a single component, keeps the number of equations and variables on an acceptable level, because complexity of one component is usually restricted. Nevertheless sparse matrix structures arising with electrical networks or jacobian matrices of mechanical parts will be used implicitly during symbolic manipulation and helps to simplify equations. Finally the symbolic model representation is used to generate the source code of a class including all variables, equations and interfaces of the component.

#### 4 Example

To demonstrate the flexibility and power of a component oriented approach a model of a 2-Arm robot was developed. The robot consists of one base component carrying one shoulder component. A drive component integrated into the shoulder takes over the rotation around its vertical axis. Two identical arm components are connected in series to the shoulder. Each arm component has mechanical interfaces to mount the arm to the shoulder or to another arm respectively as well as electrical interfaces built by power and signal connectors. Further each arm has integrated a drive component into one of its joints that serves to rotate the arm around its horizontal axes. Each drive component consists of a DC-motor with a digital encoder. It is driven by a digital servo controller via a PWM-controlled full bridge. Standard feedback control software for the 3 drive components is provided by 3 dedicated control components. The hardware of the servo controller is realized on a PC board. Coordinate transformations and feedforward motion control software for all 3 degrees of freedom is realized by a 4th control component, running on the PC processor.

#### 5 Conclusion

A very general approach to multidisciplinary modeling and simulation of mechatronic systems based on a strict component orientation was presented. An extended 3D-CAD model serves as specification target and as a single source for the generation of simulation classes for each component. Generation of multidisciplinary equations and integration of the resulting component classes in a continuous / discrete simulation environment has been accomplished and verified using a 3D model of a robot. As an example of an electromechanical component a DC-motor has been specified completely by an extended 3D-CAD model that also was used to generate the component class.

In the next steps performance of the generated equations has to be improved using preconditioning and optimized algorithms for the underlying problems e.g. LU- and QR-decomposition. Methods to specify electromechanical coupling especially in actuators will be improved and extended to magnetic systems and piezoactuators.

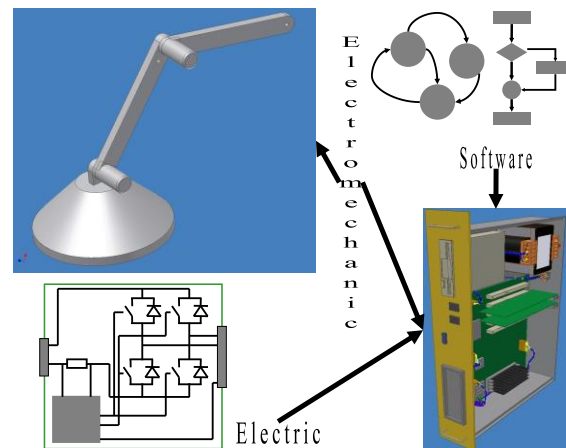


Fig. 3 Example of a robot built up from components

#### 6 References

- [1] R. Kasper, W. Koch: Active Product Models for Mechatronic Systems. 24.8.-28.8.1999, 12th International Conference on Engineering Design ICED99, Munich, 1999.
- [2] C. Scholz, W. Schiehlen, M. Krastel, R. Anderl: A Neutral Data Model for Mechatronic Systems - A Scientific Challenge. In: Preprints 1st IFAC-Conference on Mechatronic Systems (Darmstadt, 18. -20.09.2000). p. 821-830, VDI/VDE 2000, Düsseldorf, 2000.
- [3] [PSPICE9:](http://www.cadencepcb.com/products/downloads/PSpicestudent/default.asp)  
<http://www.cadencepcb.com/products/downloads/PSpicestudent/default.asp>

- [4] O. Enge: Modeling of Electromechanical Systems With Variable Structure Using the Linear Complementarity Problem, Int. J. of Applied Electromagnetics and Mechanics, 19 (2004)1-4, p. 25-29, IOS Press, 2004.
- [5] M. Tiller, P. Bowles, H. Elmqvist, D. Brück, S. E. Mattson, A. Möller, H. Olsson and M. Otter: [Detailed Vehicle Powertrain Modeling in Modelica](#). Proceedings, Modelica Workshop 2000, October, 23.-24., Lund, Sweden.
- [6] U. Ascher, H. Chin, L. Petzold, S. Reich: Stabilization of constrained mechanical systems with DAEs and invariant manifolds. The Journal of Mechanics of Structures and Machines, 23(2), p. 135-157.
- [7] R. Kasper, D. Vlasenko: Method for distributed forward dynamic simulation of constrained mechanical systems. Proceedings of ECCOMAS Thematic Conference on Advances in Computational Multibody Dynamics, Madrid, Spain, June 21-24, 2005.
- [8] R. Kasper, D. Vlasenko: Modeling and Simulation of Mechanical Parts of Mechatronic Systems. 11.-12.10.2005, "7. Magdeburg Days of Mechanical Engineering 2005", ISBN 3-929 757-83-4, Magdeburg, 2005.
- [9] R. Kasper, Y. Yefimenko: Component-oriented simulation of electrical systems in the field of mechatronic. Proceedings of the 5th EUROSIM Congress on Modeling and Simulation, ESIEE Paris, France, 2004.
- [10] R. Kasper, Y. Yefimenko: Modeling and Simulation of Electrical Parts of Mechatronic Systems. 11.-12.10.2005, "7. Magdeburg Days of Mechanical Engineering 2005", ISBN 3-929 757-83-4, Magdeburg, 2005.
- [11] R. Kasper, W. Koch, A. Wolf: Event-Based Simulation of Mixed Continuous and Discrete Components of Mechatronic Systems. 1998, 31st ISATA "Simulation, Virtual Reality and Supercomputing; Automotive Applications", proceedings, pp. 31-38, ISBN 0953257614, Düsseldorf, 1998.