DISTRIBUTED HIL SIMULATION FOR THE DESIGN OF DECENTRALIZED CONTROL STRUCTURES

Ralf Stolpe, Oliver Oberschelp

Abstract: In technical systems of the future the combination of local components from different technical disciplines with local information processing will increase. One example is a series hybrid vehicle. It consists of an auxiliary power unit, e.g,. an internal combustion engine, and a generator coupled to the crank shaft of the combustion engine, a battery and a drive unit with two electric motors. Each of these vehicle units has its own electronic control unit. Design and interplay of distributed intelligent aggregates in future technical systems will gain ever more importance. Parallelisation will be used not only for the purpose of reaching real time, but also for structuring purposes. The aim of mechatronics is the overall design of those systems. A toolset which supports the distributed HIL simulation as part of the mechatronic design cycle will be presented.

1. Mechatronic Systems

The mechatronic design cycle is characterized by taking into account and integrating mechanics, electronics, and information processing in the design process. For complex mechatronic systems a promising procedure is the decomposition of the entire system according to physically reasonable criteria. To describe the function of such combined subsystems we use the expression "mechatronic function" (Lückel 1997). Each discipline contributes its own specific function to the mechatronic function. Many technical systems boast an inherent physical parallelism, their setup showing parallel structures. Maintenance of the inherent physical parallelism in the design process means the design of parallelized mechatronic functions.

Therefore, such subsystems can be called Mechatronic Function Modules (in short: MFMs) (Honekamp 1997). An MFM consists of actuators, sensors, continuous and discrete information processing, and interfaces for mechanical coupling and information exchange.

For the hybrid vehicle two subsystems are obvious: one subsystem for the generation of electric energy, the MFM "generate energy", and another for driving the vehicle wheels, the MFM "drive wheels". The mechatronic functions "generate energy" and "drive wheels" and their assigned subsystems of the hybrid vehicle are only coupled via the battery. This characteristic allows the decoupling of the mechatronic functions. A mechanical coupling would complicate parallel decentralized control arrangements. An increasing control effort must be invested to develop a suitable decentralized control concept for these systems.

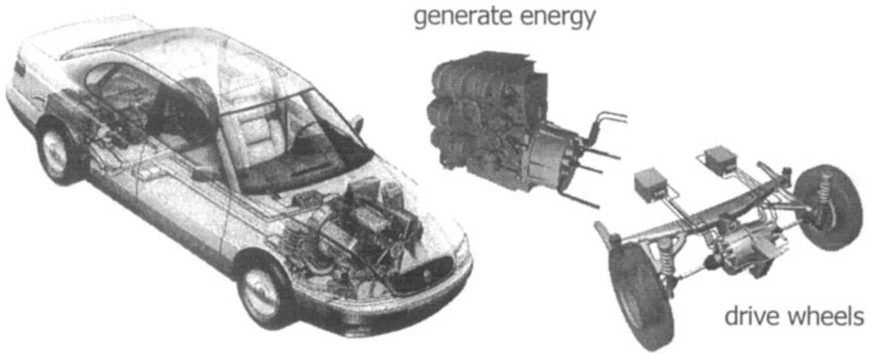

Figure 1. Functional decomposition of the series hybrid vehicle

2. IPANEMA Toolset

If you want to apply the MFM structuring approach on the realization, two features are required:
- comfortable, well-designed modelling tools which allow to arrange the components from different disciplines in a flexible way. One of the approaches to a structuring of the modelling process is represented by the MOOMo concept (Mechatronic Object-oriented Modelling) (Wolf 1998). MOOMo allows to model systems on the basis of their respective discipline-specific physical model. For realization, the concept combines elements from object-orientation and from graph theory, thus providing the condition to integrate formalisms for the derivation of the mathematical model (Hahn 1998);
- a strategy to organize parallel processes within a system and a software library supported by tools that can be used to make up distributed real-time processing. The IPANEMA toolset described in the following is designed to support this procedure.

In the mechatronic design cycle the step-by-step testing and realization of subsystems (Hardware-in-the-Loop) take an important position. The term "Distributed Hardware-in-the-Loop Simulation" means that one or more parts of a system actually exist on a testbed and that the other parts of the system are simulated on a multiprocessor hardware in real time. This requires a suitable hardware support with scalable and manageable hardware components for the tasks of computing and coupling real technical systems.

The software must also correspond to the demands on scalability and manageability. This could be achieved through a functional decomposition of the software. One part is the coupling of real technical components, mandatory for digital control and HIL. Another part is the computation, e.g., the solution of the system ODEs via numerical integration. Because of the application to different technical systems, a reusable infrastructure is important. An easy exchange of subsystems and coupling of different technical components must be possible and the supervision of real-time conditions guaranteed. Handling tasks that are not under hard real-time conditions will fall to administrative parts. There are special demands on the administrative tasks since they should be designed for parallel simulation. Important features are the simultaneous start and stop of the subsystem simulation and the simultaneous alteration of distributed parameters online. Data-logging of distributed variables is necessary for visualization and documentation purposes.

All these points were taken into account in the design of the distributed simulation platform IPANEMA (Integration Platform for Networked Mechatronic Applications) (Honekamp 1998, Stolpe 1998). IPANEMA is a library and consists of simple elements for assembling complex mechatronic applications. Parallelized computing applications become more manageable by if only a few basic element types are assembled. Furthermore, the concept will be more transparent to application engineers who like to become familar with the concept.

In order to make available defined interfaces and satisfactory portability, IPANEMA disposes of a software layer between the real-time operating system and the information processing of the mechatronic application. This software layer was introduced to encapsulate operating-system services against their application and vice versa.

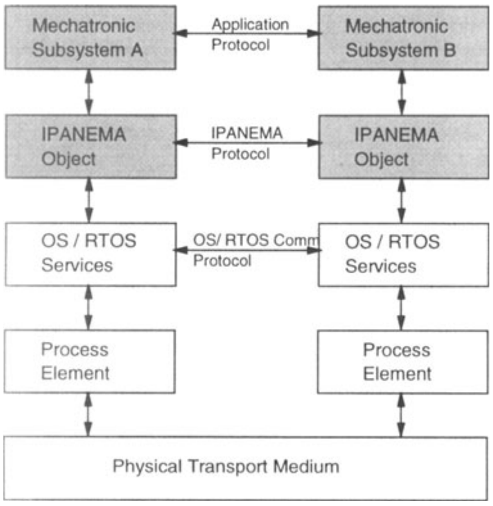

Figure 2. Software layers for the distributed mechatronic application

Additionally the application does not need to "know" specific details of the underlying basic services. This leads to a platform-independent application code and portability. The hardware supported up to now is a cluster of workstations with the operating systems Unix or NT, transputer networks with a hardware-encoded mini operating system and USAP (Power PC 604/transputer) networks (ETAS 1996). There are two different implementations on the USAP: one version uses the transputer as the owner of the control flow and the PPC with a small operating system kernel on it as a coprocessor for the computational task. The other version was designed and implemented in the METRO project (Kleinjohann 99). Therefore the operating system PEACE (Process Execution And Communication Environment) on the PPC and the real-time communication system DREAMS (Distributed Real-Time Management System) on the transputer were used (METRO 1998).

Because of the above-mentioned functional decomposition of the software, IPANEMA consists of the following objects: the first object is responsible for the processing of the application-relevant data. If implemented on a real-time hardware, such calculator processes are able to perform real-time data processing. These processes have interfaces among a set of such processes in order to exchange data needed for computation. The number of interfaces is determined by the particular application and therefore not standardized by the concept.

The second object with the name adaptor is employed for the coupling of technical components. This means the coupling of real technical components to a calculator process or to a set of calculator processes. As discussed above, real-time ability is mandatory for adaptor processes. Both adaptors and calculators are quite busy in normal applications. Therefore it seems reasonable to install additional processes in order to support the real-time tasks wherever possible. For that purpose the assistant process class was created. This process is employed for the administration of the calculators and adaptors, e.g., to start and stop the simulation and to alter parameter values online. As each calculator/adaptor has its exclusively accessible assistant, another process class is needed to moderate the set of assistants.

Figure 3 shows an architectural survey of the IPANEMA toolset with the objects mentioned. The objects are summed up in the IPANEMA library.

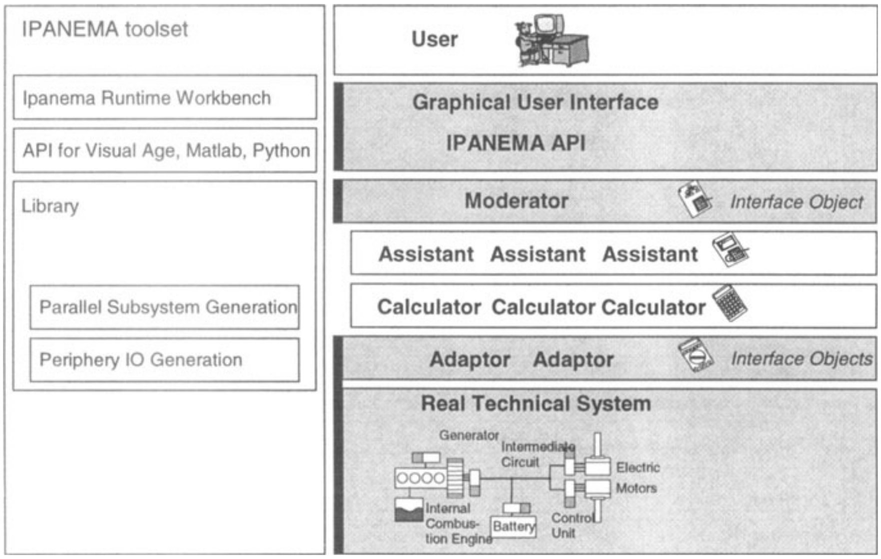

Figure 3. The Ipanema toolset

The moderator of the distributed simulation is the interface object to the user input level. On the user input level, we have an API to VisualAge, Matlab, and Python. A graphical user interface with comfortable plotting features is also part of the toolset. One important property of the moderator is net-transparency. If the user is willing to alter a system parameter or to trace variables during simulation, the moderator is able to determine the physical processor where the particular variable is stored. It is not the user's responsibility to know the physical residence of the particular variable.

Also the adaptor could be seen as an interface object to the technical system. The entire information-processing unit is distributed to a set of calculatos and adaptors. This means that each calculator processes a part of the entire system. The adaptors are generated from I/O description objects by the tool ElBaCo (see chapter 4). The calculators and adaptors are located on the level of hard real-time conditions. The calculators are generated from O-DSS (Hahn 96) by a tool concatenation for distributed simulation-code generation. This tool is still under development and will be integrated into Visual-MOOMo. These are the basic components to manage a distributed simulation.

3. Series Hybrid Vehicle

Hybrid vehicles allow to combine the advantages of the conventional drive (high performance and operating range) with those of the electric one, e.g., the opportunity of emission-free operation and regenerative braking. Figure 4 shows the basic components of a series hybrid vehicle. The hybrid vehicle consists of two electric traction motors that directly drive the two rear wheels. These motors receive their energy either from a Nickel-Cadmium battery or from a generator driven by an internal combustion engine. This aggregate is called the Auxiliary Power Unit, or just APU.

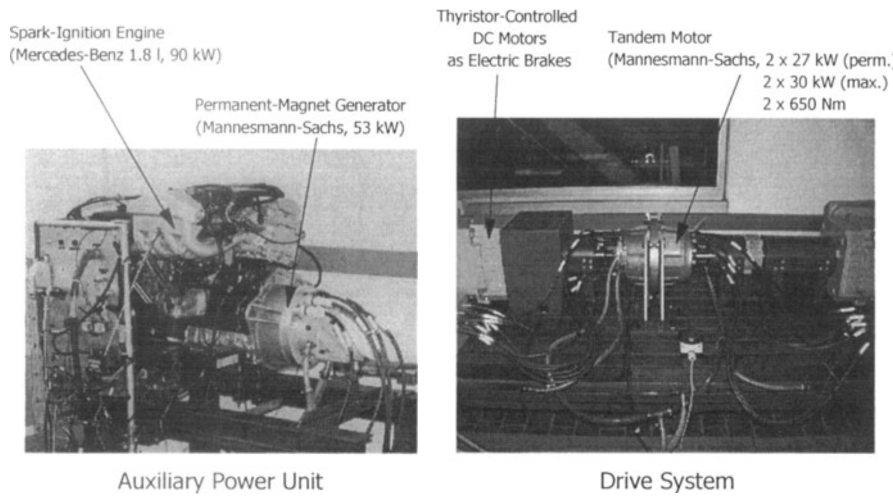

Figure 4. Arrangement of the components on the testbed

On the left-hand side of Figure 4, you see a photo of the APU with the combustion engine on our testbed. It is a Mercedes-Benz 1.8 liter spark-ignition engine from the current C-Class. It has a maximum power of 90 kW at 6000 rpm. The maximum torque is 170 Nm. The generator is a permanent-magnet synchronous machine by Mannesmann-Sachs with 53 kW. For a compact construction the rotor of the generator is directly coupled to the crank shaft of the combustion engine. The right-hand side shows the drive system of our configuration with the tandem motor in the middle and the electric brakes. The tandem motor has a maximum power of 2x30 kW and a maximum torque of 2x650 Nm. The electric brakes are inserted to act like the real vehicle with inertia (Wältermann 1998).

The testbed control is implemented on the distributed real-time PPC hardware mentioned before (Stolpe 1998). Figure 5 shows the testbed configuration with the parallelized mechatronic functions "generate energy" and "drive wheels". Several more parallel functions can still be indicated, but for more clarity, only these two will be considered in the following. The parallel control flows of the mentioned mechatronic functions are shown in Figure 5, right hand.

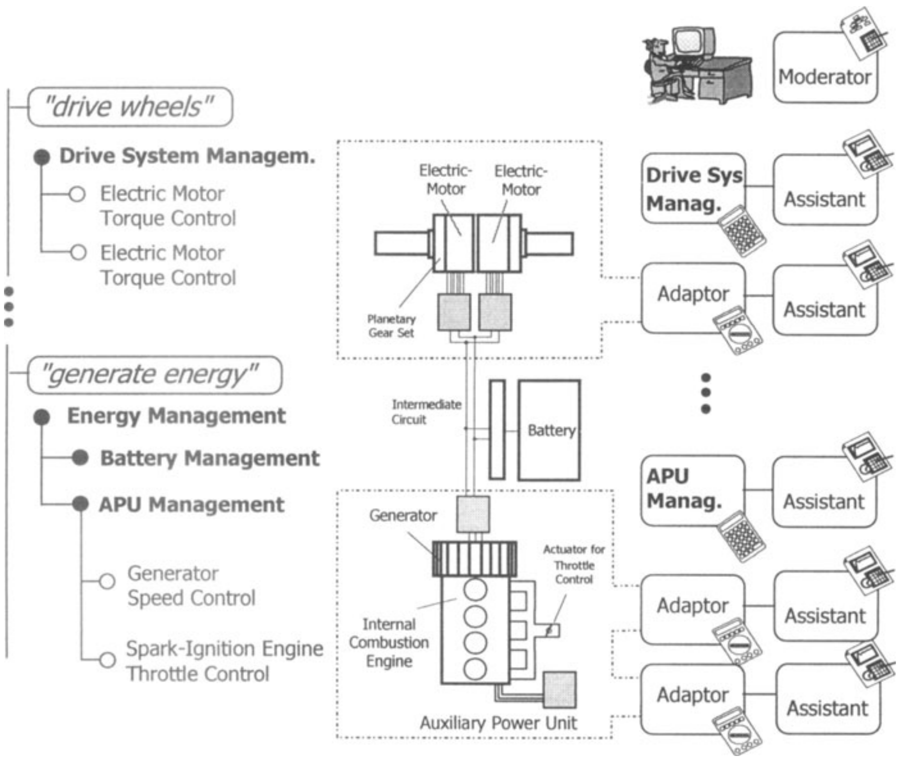

Figure 5. Testbed configuration with parallel control flows and distributed HIL simulation with the IPANEMA objects

The energy management can be divided into the auxiliary power unit management and the battery management. The former can be divided into the spark-ignition Engine control and the permanent-magnet generator control. The drive system consists of the two electric motor controllers.

In the vehicle the entire functionality will be realized by distributed controller units. The controllers working in parallel can be simulated on a parallelized hardware already at the design stage. For this and the realization of parallel mechatronic functions, the IPANEMA toolset can be used.

The energy management with the auxiliary power unit management and the battery management is assigned to one calculator and two adaptors. The adaptors are coupled to the throttle unit and to the ECU of the internal combustion engine. The drive system management is also assigned to one calculator and one adaptor. This adaptor is responsible for the access to the electric motor control units.

Already on the lowest control level, i.e., on the ECU level, you can find the speed control for the generator and the torque control of the electric motors. The task of this HIL simulation was the development of the APU management and the drive system management. Yet, this concept can also be applied to the development of the low-level control functions, e.g., the electric motor torque control. After testing and optimizing the drive system management the calculator with the developed control algorithm and the assigned adaptor could be ported to an ECU of their own.

4. Coupling of Periphery Hardware

The transition of a simple simulation to a Hardware-In-The-Loop simulation requires the embedding of real technical systems into the simulation environment. As a first step the drive system management and the electric motors with their corresponding controllers can be simulated completely. To link the technical process, one or several simulated subsystems can be extracted from the entire system and replaced by a real system. In this case, this means for the simulation platform IPANEMA that a calculator process has to be replaced by an adaptor process. However, this is only a simplified representation. In reality, the corresponding peripheral boards must of course be equipped for the measurement and excitation of the real process.

With this additional hardware higher demands on the coupling process arise than on the calculator. Calculators could be generated automatically from the model description. For the generation of the adaptor code, additional information about the peripheral boards and the mounted technical process is required. These information are usually not included in the physical model description. Answers are required for the following questions:

- Which peripheral board is employed?
- Which connection of the board is employed?
- To which physical quantity does the measured value correspond?

In the IPANEMA toolset these informations are summed up in one or more hardware-coupling descriptions (HCD).

However, even further information is required for embedding into the IPANEMA topology. The main topics are the following:

- Which channels to which calculators are necessary?
- Which measured and controlled value is transmitted via which channel?

These are the coupling informations, which are summarized in the process-coupling description (PCD).

5. Element-based Compiler for Argument-depending Problems of Code Generation

Since the adaptor code depends strongly on the inserted peripheral boards and, what is more, since these may change often, a flexible, fast, adaptable code generation is required. This demand is met by an instrument developed within the context of the METRO project. It is called ElBaCo.

The name ElBaCo stands for **el**ement-**ba**sed **co**mpiler for argument-depending problems of code generation (Oberschelp 1998).

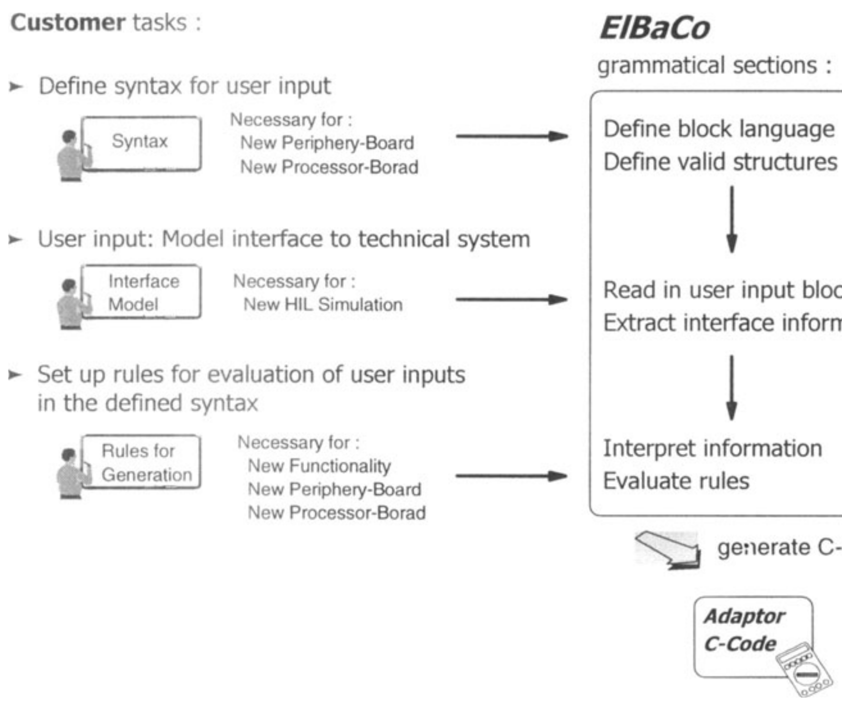

Figure 6. The working method of ElBaCo

ElBaCo is a simple tool for the generation of ASCII source code. It uses a configurable block language which is assembled from predefined grammatical elements as input language. The relevant information is extracted by an interpreter from the input language and code created according to it.

ElBaCo puts moderate demands on the computer hardware and the operating system and is easily portable onto different platforms since it was implemented completely in ANSI-C. It is a flexible compiler since its input language is adaptable.

The syntax is defined by selection from fixed grammatical constructs and the definition of parameters, variables, and constants. These information are stored in a text file which represents the language specification. The interface information contained in the user input file is structured by the syntax and transformed to a data structure. The information in the data structure could be evaluated by an integrated interpreter and used for the generation of code. The syntax definition and the interpretative evaluation rules are written only once for the compilation tasks. The task is finished by an expert for the code to be generated. The user will only use the syntax definition and evaluation. See Figure 6 for the user tasks and the grammatical sections of ElBaCo.

ElBaCo is implemented by means of a special instrument for the generation of compilers. This programming environment of the name ELI (Environment for Language Implementation) is a joint effort of the University of Paderborn, the University of Colorado (USA), and the James Cook university (Australia) (ELI 90).

It offers complete solutions for commonly encounted language-implementation subtasks. ELI contains libraries of reusable specifications, allowing to elaborate high-quality implementations from simple problem descriptions. The environment is problem-oriented instead of tool-oriented, which is the common working method in other tools. The user describes the problem to be solved and then ELI automatically employs the tools and components needed for that particular problem. The compiler construction set is an integrated system, and as a result it generates a complete set of C modules that leads to an easy portability to different platforms.

6. Conclusion

The recognition and installation of parallel control flows are an essential starting point for the design of decentralized control concepts of mechatronic systems. Installation of those subsystems is often physically motivated. This was illustrated by the testbed of the series hybrid drive train. For the realization of the distributed HIL simulation, the IPANEMA toolset was used. The user is supported by the flexible code-generating tool ElBaCo for the coupling of real components on the testbed, by an interpretative user interface and also by a comfortable graphical user interface.

References

dSPACE GmbH. 1991. *DSP-CITpro Hardware.* Paderborn, Germany.

ETAS GmbH & Co. KG. 1996. *USAP Transputer Module.* ETAS PC604-2000 V2 Documentation. Stuttgart, Germany.

Hahn M., Lückel J., Naumann R., Rasche R. "Ein Objektmodell für den Mechatronikentwurf." *ASIM'98. 12. Symposium Simulationstechnik* (Zürich, Switzerland., Sept. 15-18, 1998).

Hahn M., Meier-Noe U. 1996. "Classification in the Object-Oriented Modelling Language Objective-DSS, Exemplified by Vehicle Suspensions." *International Symposium on Computer-Aided Control.* IEEE. (Dearborn, MI, USA, Sept. 15-17, 1996).

Honekamp U. 1998. *IPANEMA - Verteilte Echtzeit-Informationsverarbeitung in mechatronischen Systemen.* Diss., Mechatronics Laboratory Paderborn, Germany.

Jäker K.-P., Klingebiel P., Lefarth U., Lückel J., Richert J., Rutz R. 1991. "Tool Integration by Way of a Computer-Aided Mechatronics Laboratory (CAMeL)." *5th IFAC/IMACS Symposium on CADCS 91* (Swansea, UK, 1991).

Kleinjohann B., Rammig F., Schmitfranz B.-H., Schröder-Preikschat W., Stolpe R., Wältermann P. 1999: "Parallel Computing for the Design and the Implementation of Mechatronic Systems: Lessons Learned from the METRO Project." *3rd International HEINZ NIXDORF SYMPOSIUM on Mechatronics and Advanced Motion Control* (Paderborn, Germany, May 27 – 28, 1999).

Lückel J., Wallaschek J. 1997. "Functional Modelling and Simulation in Mechanical Design and Mechatronics." *2nd MATHMOD* (Vienna, Austria, Feb. 5-7, 1997).

Honekamp U., Stolpe R., Naumann R., Lückel J. 1997. "Structuring Approach for Complex Mechatronic Systems." *30th ISATA Conference* (Florence, Italy, June 16-19, 1997).

METRO 1998. "Einsatz massiv paralleler Rechner beim Entwurf und der Realisierung komplexer mechatronischer Systeme." Abschlußbericht BMBF Verbundprojekt METRO. Daimler-Benz AG, ETAS GmbH, GMD-FIRST, Mannesmann-Sachs AG, TU Chemnitz (IfM), Uni-GH Paderborn (C-LAB, MLaP, HNI).

Oberschelp O. 1998. "Design and Implementation of a Flexible Code Generator to Couple Technical Processes in Hardware-in-the-Loop Simulations" (in German). Internal Paper, University of Paderborn, Mechatronics Laboratory Paderborn, Germany.

Stolpe R., Zanella M.C. 1998. "A Distributed Hardware-in-the-Loop Simulation Environment in Use on a Testbed of a Series Hybrid Drive." *ESM 98, 12th European Simulation Multiconference* (Manchester, UK, June 16-19, 1998).

Stolpe R. 1997. "Verteilte Simulation und Realisierung mechatronischer Systeme am Beispiel einer hybriden Roboterregelung". *PEARL 97, Workshop über Realzeitsysteme* (Boppard, Germany, Nov. 27-28, 1997).

ELI 1990. Übersetzerbau, Band 3.3. Handbuch der Informatik, R. Oldenburg Verlag, München/ Wien, 1990.

Wältermann P., Neuendorf N. 1998. "A Testbed for the Design and the Optimisation of a Series Hybrid Drive Train". *International Conference on Combustion Engines and Hybrid Vehicles* (London, UK, April 28-30, 1998).

Wolf M. 1998. "Visual MOOMo – A Graphical Object-Oriented Modelling Environment for the Design of Mechatronic Systems". MECHATRONICS 98, 6th UK Mechatronics Forum International Conference (Mount Billingden, Sweden, Sept. 9-11, 1998).