Mastering Complexity

A partnership with Daimler Chrysler for quality assurance in automotive embedded systems

When somebody gets into one of today's new cars, dozens of electronic assistants get busy helping them to drive, including such systems as adaptive cruise control (Distronic), the Electronic Stability Program (ESP) and the parking assistant. These systems make driving easier, safer and more efficient, yet the wide array of networked electronic systems has brought a new dimension of complexity to the development of new car models. A modern control device is a highly complex, embedded, software-based system that must function reliably in a network of 30–80 other such devices, depending on the type of car. The more complex the overall in-car system is, the more challenging it becomes to ensure smooth interoperability of all individual parts. A further difficulty is that individual control devices are developed by different automotive suppliers who obviously cannot test run them in their final configuration. A single class of car often involves hundreds of software developers producing something in the region of ten million lines of programming code. Thus ensuring faultless systems integration is a major challenge for the automaker. DaimlerChrysler has set up a "Zero Fault Initiative" whose rules apply to the whole Group and all of its suppliers. One of its key features is a systematic test process for hardware and software. The well-known crash testing with dummies in the finished car is now more by way of a final touch at the end of the quality assurance process. Testing and trial running of single automotive components now begins in simulation before they even exist in the real world.

> „We first got to know Fraunhofer FIRST when the two of us collaborated on publicly funded projects. The Embedded Systems department has the kind of expertise we're looking for so entering into partnership was only logical."
>
> Dr. Joachim Wegener, Project Manager at DaimlerChrysler

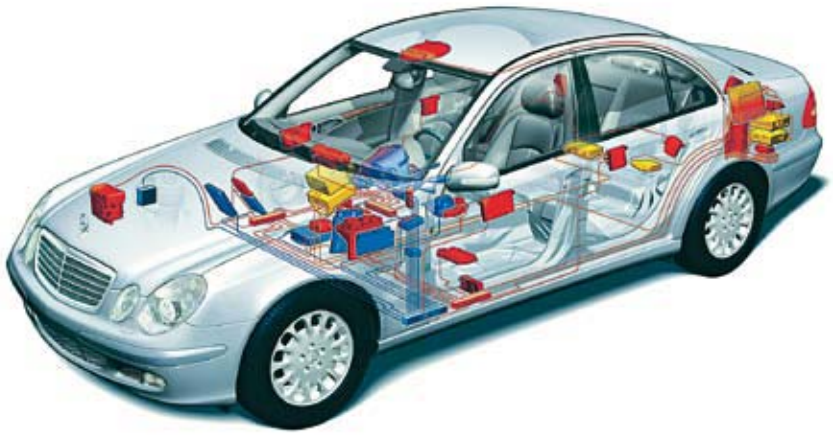
In December 2003 DaimlerChrysler's Berlin Research Center entered into a three year framework agreement with Fraunhofer FIRST's Embedded Systems department to drive forward optimization of testing for software-based control devices. Within this agreement Fraunhofer researchers worked on a total of eight projects together with the Software Technology Research Laboratory's Unit for Methods and Tools. Researchers in this unit act as in-house consultants for the DaimlerChrysler Group. They support various Group departments across the whole software development cycle from definition of requirements and systems modeling to definition and administration of the tests.

The project manager for DaimlerChrysler, Dr. Joachim Wegener, explains why the Group chose Fraunhofer FIRST as a research partner, "We first got to know Fraunhofer FIRST when the two of us collaborated on publicly funded projects. At the same time as the framework agreement, we were also working on the IMMOS project – Integrated Method for the Model-Based Development of Automotive Control Units – funded by the Federal Ministry of Education and Research. A lot of synergy effects were generated here. The Embedded Systems department has the kind of expertise we are looking for so entering into partnership was only logical."

As the project manager at Fraunhofer FIRST, Prof. Dr. Holger Schlingloff, sees the partnership, "The automotive sector is of key importance to us in the Embedded Systems department. No other industry produces so many software-based innovations in such a short time. I got a great deal of satisfaction in working on quality assurance of control devices that were perhaps later destined to go into serial production for in-car use. The general agreement had the advantage of substantially reducing the typical administrative

Quality assurance. Control device software is first tested with virtual models before it gets a trial run in real world prototypes

Resilient networks. Nowadays there are from 30–80 control devices in a car, depending on the model



outlay involved in the start-up of a new project, which meant that as Project Manager I could concentrate much more on actual project work."

Model-based development of control device software first became widespread in the automotive industry in the mid 1990s. Traditional methods of software development are based on the catalogue of specifications and requirements, a written specification of software requirements, which is used to draw up the program and port it in the hardware. Model-based development, on the other hand, uses the requirements analysis as a basis for modeling and simulating the embedded systems software on the computer. A systems model of the control device to be developed is made with a description of all the control, regulation and monitoring functions to be implemented, and an appropriate environment model is constructed. This system model is then run in a real time environment simulation on powerful computers to test its functional behavior. During development work the original system model is gradually refined into an implementation model whilst being subject to continuous testing at each stage. If the control device algorithm realized in the implementation model fulfils all the performance specifications, a model compiler can be used to automatically generate from it the major part of the software for the definitive version of the control device.

Model-based development has some obvious advantages. Testing at the model stage saves time and money whilst at the same time test cases can now be carried out that would be impossible to run in real life for a variety of reasons, including safety constraints. Quality is also improved by precluding any variation between the draft and actual program code. What's more, in practical terms it has been shown that even greater time savings can be made in the software development cycle.

In their joint projects researchers from Fraunhofer FIRST and DaimlerChrysler have refined model-based development yet further. They produced simulation models of control devices using the MATLAB/Simulink modeling tool, evaluated various models, and designed and administered modeling guidelines, producing a comprehensive set of online directives for which Fraunhofer researchers also developed a web interface and user administration. Automatic code generation in model-based development requires new methods of safeguarding code and testing – and also makes them possible. It is also a great advantage that model-based development can engineer automatic methods of test generation. The researchers evaluated the automatic code generation and testing methods and inspected code generators. But obviously evaluation of the control device software is not just limited to its virtual model. Once the simulation and code generation stages are successfully completed, the real individual control devices are embedded in Hardware-in-the-Loop testbeds – in other words the control device is integrated in an environment simulation where it is tested. The "loop" here refers to the cyclical streaming of data from the environment model into the control device and back again. Several control devices can be integrated in the testbed at the same time as a reliable means of validating their interaction. To this end researchers at DaimlerChrysler and Fraunhofer FIRST developed a test evaluation language and report generation system for Time Partition Testing (TPT), a testing tool developed by DaimlerChrysler for embedded systems which supports the whole process from specification and implementation through to evaluation and test documentation. Test implementation in the Hardware-in-the-Loop testbeds was in real time. DaimlerChrysler and Fraunhofer FIRST shall continue their collaboration over the next three years. One joint research project now underway is the EvoTest project funded by the European Union and researching evolutionary test generation. The project aims at the automatic generation of test suites using the laws of evolutionary biology. Biological mechanisms like selection and mutation shall be utilized to provide optimum test cover-

age. As Dr. Wegener comments, "I am delighted that our collaborative efforts will continue over the next few years in the EvoTest project. Even if our research and pre-development department is moving to Böblingen, DaimlerChrysler still attaches a great deal of importance to Berlin – as can be seen by its founding the DCAITI, DaimlerChrysler Automotive Information Technology Institute, in the city where it will cooperate with the TU Berlin and extra-mural research institutes like the Fraunhofer-Gesellschaft. I would not want to miss such extra-mural expertise as it gives us important new impulses for our daily work."

> "The automotive sector is of key importance to us in the Embedded Systems department. No other industry produces so many software-based innovations in such a short time."
>
> Prof. Dr. Holger Schlingloff, Project Manager at FIRST

The V-Model and Model Based Development

Originally developed in Germany for use in government projects, the V-Model (Vorgehensmodell) is now an internationally accepted standard for describing the activities and artifacts in software development. It has been continually revised and extended since 1986 with the most widespread version dating from 1997 and the new V-Model XT (Extreme Tailoring) released in February 2005. The V-Model basically distinguishes various levels of abstraction in terms of the system description, user layer, internal architecture, composition of modules and, finally, the programming code of individual functions. In systems development each level has a constructive phase (requirements analysis, architecture, design and implementation), and an analytical phase (functions and module testing, integration and deployment). Although the V-Model does not require a strict time sequencing of activities, quality assurance measures for any phase typically commence after completion of those of its underlying level. This means that many errors are only identified later on, which leads to high remedial costs. Another problem is that the various levels of abstraction use different means of description which leads to inconsistencies and the incompatibility of artifacts.

Model-based development seeks to overcome these problems by using uniform modeling formalisms and consistent validation activities. Frequently used modeling languages are Simulink/StateFlow and diagrams of the UML-2 Unified Modeling Language. First a platform-independent model is produced from the description of system requirements which is then refined over successive stages into an implementation model from which the code can be automatically generated. Testing and trial running the model takes place in tandem with the refining process. Special modeling and testing tools enable the various system model refining stages to be simulated together with modeling of user and environment behavior. This allows statements to be made early on about the correctness and appropriateness of systems behavior. Such benefits are leading to ever wider use of model-based development as a variation of the V-Model in a broad range of application environments.