Software Technology in an Automotive Company - Major Challenges

Klaus Grimm
DaimlerChrysler AG, Research and Technology
Alt-Moabit 96A, 10559 Berlin, Germany
klaus.grimm@daimlerchrysler.com

Abstract

The automotive industry is one of the sectors which has been affected significantly by the industrial software revolution during the last few years. Competitive challenges, which are of major relevance for the growing number of software-based automotive innovations, are presented by a dramatic growth in system complexity, rising time and cost pressure, and high quality demands.

All these requirements lead to a number of software-related challenges which will be of significant competitive importance for the automotive industry in the future. Asides from the question of whether an automotive company decides to develop software in-house or whether this is carried out by suppliers, every automotive key player has to hold or to build up specific software competencies. The most important competence areas are the software development process, software quality management including supplier cooperation, the overall architecture of the in-vehicle software, as well as the ability to specify, to integrate and to test the system.

1. Introduction

The importance of software is increasing dramatically in nearly all sectors of industry. In the area of business administration, software has become an indispensable core technology. However, more and more products and innovations in most technical fields are also based on software. Traditional application fields are aeronautics, and space and defence systems in which software has already played an important role for a few decades. During the past few years software has also become established in many industrial sectors and products which had been determined by mechanical and electrical engineering traditionally. Within the automotive industry, for example, more and more innovations are based on electronics and software to enhance the safety of the vehicles but also to improve the comfort of the passengers and to reduce consumption and emission.

The increasing share of software in the car is accompanied by certain trends and requirements, such as rising time and cost pressure and high quality demands. These trends are presented in the next section, followed by a description of general problem areas in section 3 and technical challenges facing the automotive company in section 4. The core competencies required to meet these challenges are outlined in section 5. Section 6 provides a survey of current activities and results gained by automotive software technology research at DaimlerChrysler. The article closes with a short summary and the outlook for future work.

2. Software-related trends

The share of software within a vehicle is steadily growing. At the same time more and more competitive differentiation is realized by means of software-based functions. Current examples are intelligent powertrain control, the Electronic Stability Program ESP, and the active distance control DISTRONIC which recently entered the market as a comfort feature of the new Mercedes-Benz S-Class. DaimlerChrysler experts estimate that 80 percent of all future automotive innovations will be driven by electronics, 90 percent thereof by software. Leading automotive companies will not be able to reach their future strategic goals such as accident-free driving or significant reduction of consumption and emission without software. This will further increase the amount and importance of software in the automotive industry.

The rising share of software in the vehicle is accompanied by a growing complexity of the systems. In the past every single control device had to fulfil a single dedicated task such as electronic injection or airbag control, and the devices were simply connected by point-to-point connections if there was any direct link at all. In the early nineties, more and more functions, such as the Anti Slip Control, were developed which could only be realized by the cooperation of different sub-functions and control devices coupled by a single network like the CAN (Controller Area Network) bus. In the late nineties the engineers started to couple control units using multiple networks such as CAN bus and MOST (Media Oriented System Transport) bus. Thereby, functions were designed which e.g. required the combination of vehicle dynamics and navigation information. Nowadays, functions such as Teleaid are developed which comprise several networks and which are also coupled to the outer-car world via radio link. Teleaid provides the automatic alert of an emergency service in case of an accident detected by the airbag control. A current example for the high degree of software-related complexity of today's cars, shown in figure 1, is provided by the current S-Class of Mercedes-Benz which contains more than 50 controllers, more than 600,000 lines of code, three different bus systems and approximately 150 messages and 600 signals.

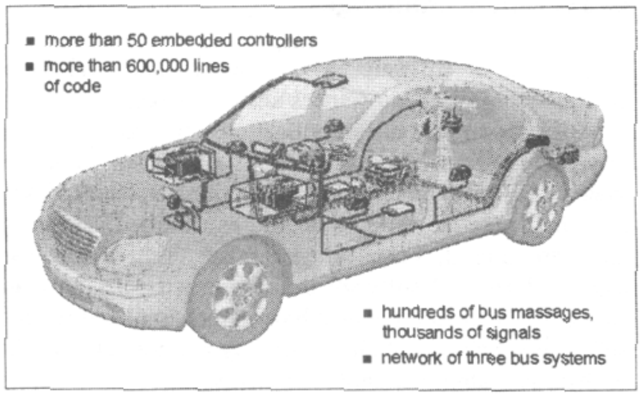
■ more than 50 embedded controllers
■ more than 600,000 lines of code
■ hundreds of bus massages, thousands of signals
■ network of three bus systems

Figure 1. The complexity of the current Mercedes-Benz S-Class

The growing use of software-based automotive functions has led to software gaining a significant impact on the quality of the car. Nowadays, automobiles have to fulfil high software quality demands in order to satisfy the customer and to be in line with legal requirements. The most important quality criteria are reliability, availability, and safety. As more and more interfaces to the driver are realized by software, user friendliness and ease of use also become important quality demands. A mature software development process including cooperation with suppliers, a comprehensive quality management and powerful methods and tools is required to fulfil the high quality requirements.

The automotive industry has to face enormous time and cost pressure. Particularly premium class producing companies have to launch their innovations quickly and to achieve a short time to market in order to gain competitive advantages and to underline their premium image. To the same extent as innovations are driven by software, the share of development cost, as well as warranty and goodwill costs incurred by software, increases. Approximately 25 percent of the cost of the current Mercedes-Benz S-Class is already incurred by electronics, including software. As shown in figure 2, this proportion will rise by up to 40 percent during the next ten years due to additional safety and comfort systems as well as modern entertainment and infotainment services. On the other hand, the use of software offers a much higher flexibility than can be achieved by realizing innovations with conventional automotive technologies only. Especially appropriate software architectures allow fast innovations and a high degree of flexibility and reuse as well as leading to a significantly more efficient software development.

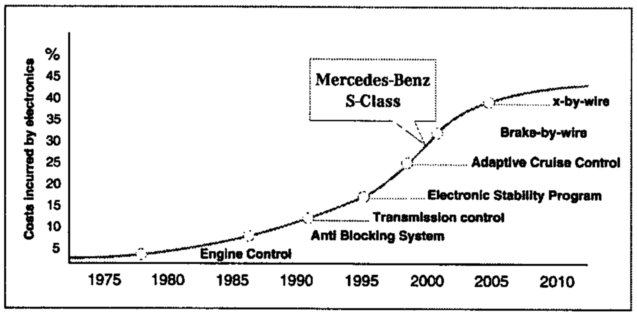

Figure 2. Proportion of cost incurred by electronics

In contrast to the traditional technologies used in the automotive industry, such as mechanical and electrical engineering, the use of software is subject to a fast succession of technologies. These fast changes hold the danger that software-based innovations become obsolete early and that technologies which were used to realize the innovations are replaced quickly by more powerful ones. Particularly in the field of telematics, innovation cycles are much shorter than the current development cycles of new vehicle models. In order to safeguard innovations and corresponding investments as long as possible, the software technologies to be used in automotive development have to be selected very carefully. Furthermore, the software-based system, its architecture and its interfaces have to be designed openly and flexibly enough to follow the short telematics innovation cycles and to facilitate the easy replacement of old-fashioned telematics equipment during the life-time of a specific vehicle model.

3. General problem areas

Regarding software-based innovations in vehicles, the depth of the added value created by software at the automotive company represents an important aspect. Currently, a lot of software development is undertaken by suppliers. But more and more automotive companies have decided to develop some of those parts of their software in-house which allow competitive differentiation. Whether an automotive company develops its software in-house or not, it has to face several software-related challenges.

The control of the increasing complexity of the systems represents a core challenge. The complexity problem affects different phases and elements of system development, especially the specification of the requirements, the design and the architecture of the systems as well as the integration and testing of the sub-functions and sub-systems. Complexity control becomes even more difficult if several suppliers are involved in the development of the system.

An indispensable prerequisite for the efficient development of dependable software-based systems is a mature requirements specification. In practice, this objective is very difficult to achieve, particularly because not all requirements are clear or stable right from the start, and a lot of changes in the requirements become necessary in the course of development due to technical or customer-related reasons. In-house experience has shown that, on average, more than 40 percent of errors which occur during the use of an automotive software-based function are attributable to requirements errors caused by immature specifications.

The development of electronic control units for automotive use becomes even more complicated if it takes place in cooperation between an automotive company and suppliers. In many projects the hardware and major parts of the software are developed by suppliers, while software functions which allow competitive differentiation are built by the automotive company. In this case, even more emphasis has to be put on the maturity of the requirements specification and the integration and testing aspects. The entire development process, as well as quality management, has to include the suppliers' part and the aspects of distributed development.

Due to the high time and cost pressure, the productivity of the software development is of great importance for an automotive company. One objective within this context is the increasing reuse of development results such as requirements, models, functions, and software components. Today, the degree of reuse is low. There is a lack of constructive prerequisites such as an open and flexible software architecture to facilitate reuse and integration of existing code into newly built systems. As a consequence, some functions are built nearly from scratch for each vehicle model.

The efficiency of the software development and the quality of the software-based systems can be improved significantly by the use of appropriate methods and tools. Nowadays, methodological and tool support exists for different single tasks and phases of development. Unfortunately, there is no chain of methods and tools which seamlessly support the whole development cycle from requirements specification to integration and testing. As a consequence, gaps between phases have to be covered manually, which is an error-prone and time-consuming exercise. Furthermore, tracing of information such as requirements or design decisions and change management are awkward, involved and costly. This missing seamlessness of processes, methods and tools is already an unavoidable problem in the field of software technology. Its dimension increases if one considers the interdependencies of software and other core disciplines of automotive development such as electrical and mechanical engineering. Currently, each discipline has its own processes, methods and tools. The lack of harmonization between these worlds causes many expensive errors or changes in the course of the development of a new vehicle model.

4. Technical challenges

The use of software in vehicles is characterized by certain technical circumstances and challenges.

Apart from telematics applications such as navigation, in-vehicle software generally forms part of an embedded system. This requires a close connection between software and hardware development or at least, if standard hardware is used, a consequent consideration of hardware aspects during all software development stages. In many cases, hardware components have to be simulated in early software development phases as they are not yet available or their availability is limited or costly.

Very often embedded systems in a car have to fulfil hard real-time requirements. For instance, an airbag controller only works correctly if it reacts within a precisely defined period of time if an accident is detected. In practice, both design and testing of real-time properties are complicated tasks and require a lot of modeling, simulation, and practical trials.

An additional challenge is provided by the growing complexity, namely the high degree of connectivity between the electronic control units. Not only the design of the overall system, including an appropriate, open and flexible software architecture, but also the integration and testing of the system are complex and elaborate tasks.

The use of embedded systems in the vehicle is accompanied by real-time requirements and also by limited capacity of storage. This limitation is often the result of space or cost restrictions. Due to the large number of cars which are to be equipped with certain components or electronic control units the cost of each single unit has a perceptible influence on the profit margin of a vehicle model.

As software is used to realize more than simple comfort functions, high reliability and safety requirements have to be met by the systems. This requires a proper and mature development process and the use of appropriate state-of-the-art methods in all development phases.

Suitable and powerful constructive and analytical measures have to be taken to ensure and to prove the required system safety and reliability, taking into account all relevant software aspects.

The increasing use of software-based driver interfaces and new MMI-concepts makes strong demands on the ergonomics of the system. The system has to be easily and intuitively usable by the driver without causing any dangerous situations, for instance by distracting him from the traffic.

5. Required technological core competencies

The discussion of the general trends and problem areas and the more detailed technical challenges show that an automotive company has to have or to develop certain software-related competencies. In most cases, the importance of these competencies is independent of whether the actual software coding takes place in-house or is carried out by suppliers.

Requirements engineering is the most important competency to avoid errors in early development stages and to reduce cost caused by requirements changes. Requirements engineering comprises appropriate techniques for elicitation, specification, modeling, and management of requirements. In particular, requirements tracing and change management are necessary to control the technical and economic impact of requirements or design changes.

As efficiency and time to market are of high competitive importance, the automotive company has to be able to define, analyze, assess and improve its development process systematically. A high degree of process maturity does not automatically guarantee high efficiency and quality, but it is an important prerequisite for both the manufacturer and the supplier.

The software architecture is a central element of the system of software-based innovations. Hence, the automotive company has to be able to design and maintain an open and extendable overall architecture and corresponding interfaces in order to facilitate the fast integration of innovations and the flexible replacement or reuse of software components and control devices.

Systematic testing is indispensable in order to detect errors as early as possible. This applies to software developed in-house as well as – to an even larger extent – to software supplied by partner companies. Special emphasis has to be placed on the aspect of integration of different components from different sources. In early development stages appropriate analysis and testing techniques, such as reviews, inspections and simulations, have to be applied to specifications and design models. Apart from functional behavior, performance aspects such as real-time system properties also have to be considered.

Software quality management has to cover the entire development process including the aspects of software acquisition and cooperation with suppliers. In order to avoid errors occurring due to method or tool inconsistencies and corresponding manual work, the automotive company has to be able to integrate appropriate methods and tools to achieve a seamless chain of computer support. Finally, usability engineering is an important competency to provide user-friendly software-based driver interfaces. The aspect of usability has to be taken into account throughout the whole development process starting with the specification of requirements in order to avoid late and, hence, expensive changes.

An additional requirement is caused by the growing complexity of the systems accompanied by the combination of functions of different areas such as vehicle dynamics, powertrain management, and telematics. As a consequence, appropriate architectures, as well as specification, modeling and testing techniques, are required which reconcile the different scientific disciplines involved, such as software engineering, control engineering and communication technology.

Model-based development is an essential technological competency if the automotive company carries out in-house software development. It facilitates early simulation and error detection as well as automatic generation of target code. Thereby, development becomes more efficient and the quality of the system can be improved significantly.

6. Selected research activities and results

In order to take the current trends and the resulting challenges and requirements into account, a Software Technology Research Lab was established at DaimlerChrysler in 1998. The work is divided into the areas of Process Organization, Software Architectures, as well as Methods and Tools and serves all business units of DC. The most important application areas are placed within the automotive sector. Both for the Passenger Cars Division as well as for Commercial Vehicles innovative solutions are being developed in the fields of vehicle dynamics, powertrain, comfort electronics, telematics, and sales. In addition, specific software technology topics are worked on for subsidiary companies, in particular for the European Aerospace and Defense Company EADS.

6.1. Processes

Based on established process standards such as the V-model propagated by the German Ministry of Defense, several similar process models have been developed at different automotive companies. Recently, a process-oriented software quality management handbook was released at DaimlerChrysler providing clear guidelines for the different software development steps. Experience has shown that specific reference models are required for the different automotive application areas such as vehicle dynamics, powertrain, body electronics, and telematics. The differing requirements are mainly related to the duration of the respective development cycles, to the degree of dependence from vehicle prototypes and trial rides, and to the involvement of the end-user. For instance, in the field of vehicle dynamics software development is closely coupled with the development of vehicle prototypes, whereas the integration of many cooperating, complex subsystems and the direct involvement of the driver is of greater importance in the area of body electronics.

In projects which involve not only the development of pure system functionality but also the design of user interfaces, e.g. for telematics and body electronics applications, the reference models are extended by the sub-process of "user-centric design and development". This sub-process aims at the seamless integration of all usability aspects into the entire software development process [1].

Major elements of DaimlerChrysler's research into requirements engineering are described in [2]. The work is focused on methodological aspects such as elicitation, specification, and refinement of requirements. It is complemented by corresponding analytical measures such as early simulation of requirements and the derivation of test cases from specifications. Requirements management represents another important area of work comprising the structure, administration, and tracing of requirements across the entire system life cycle. An essential result of this research work is a generic information model for requirements management which has been tailored to several automotive and aerospace applications. In order to provide powerful computer support the model has been mapped to different existing requirements management tools such as DOORS.

The specific problems occurring during cooperation with suppliers are tackled in a project focused on software acquisition management [3]. An appropriate process model has been developed which comprises the organization, the methodological support and the establishment of an efficient process for cooperation with suppliers. Emphasis is placed on quality and risk management on the part of the customer as well as on interaction with the corresponding procedures on the part of the supplier. Furthermore, a specific technique has been defined for the assessment of suppliers and their products.

6.2. Methods and tools

Major progress has been made in the area of methods and tools for automotive software development during the last few years. Tools for model-based development have found their way into the automotive development departments, and computer support has also been established in the areas of requirements management and testing.

At DaimlerChrysler Research significant effort has been made during the last five years to improve the methodological foundations of automotive software development, as well as the integration of tools and systematic and efficient testing. Test tools in particular contribute to a significant increase in test efficiency.

The design of test cases has a decisive impact on the quality of a test. Apart from common test criteria related to the coverage of the program code the required and specified functions of the system for the test must be considered. The Classification-Tree Method [4] and the Classification-Tree Editor CTE [5] have been developed by DaimlerChrysler Research in order to guarantee a systematic and comprehensible approach to the design of functional test cases. Both method and tool have been tried out successfully in numerous DaimlerChrysler projects, and have become established in development standards and environments of different business units and divisions. An extension of the approach is currently under development which explicitly considers the aspects of embedded systems with continuous input sequences.

The test system TESSY provides computer support for the testing of modules written in C [5,6]. It covers all testing activities from test case design to test execution, test evaluation, and test documentation. The functional test case design is supported by means of the Classification-Tree Editor CTE as an integral part of TESSY. Furthermore, host and target testing, automatic regression testing, and completely automated run-time testing are provided by the system. TESSY is being successfully used in different divisions of DaimlerChrysler, for instance for testing motor control units and powertrain management.

The tool MTest provides a powerful computer support for testing on the level of system models [7]. Thereby, errors made when modeling software-based functions can be detected early, i.e. before the actual coding process starts. Test cases defined for testing the model can easily be reused in later phases for testing software components and the completed system. MTest is based on the modeling tool Matlab/Simulink and is mainly used in the area of vehicle dynamics at present.

In the area of embedded systems, the correct behavior of the system depends on the fulfillment of functional as well as of real-time requirements. At DaimlerChrysler Research a new approach to real-time testing has been developed which is based on the use of evolutionary algorithms [8]. It has been successfully tried out in the areas of motor electronics and airbag control. The objective of the evolutionary test is to detect errors within the run-time behavior of the system. Input situations are searched for which lead to extremely short or extremely long execution times causing the system to fail the specified timing requirements. The search for these input situations is regarded as an optimization problem to be tackled by means of evolutionary algorithms. The target function of the optimization is provided by the execution times. The input domain of the system defines the search space.

6.3. Software architectures

As the architecture of the software is an essential element of the system ability of an automotive company, DaimlerChrysler Research currently works on both the design and the evaluation of in-vehicle software architectures. In the course of an evaluation an existing architecture or a draft version is analyzed with regard to different criteria in order to check whether or not given requirements are met, for instance concerning complexity or extendibility [9]. In general, the development of an architecture is based on a control circle between design and evaluation. In particular, the interplay between these tasks becomes very important if changes occur in the course of the software development. As the software architecture in a vehicle also depends on the mechanical and the electrical architecture there is also a mutual interdependency between the physical construction of the vehicle and the architecture of the software.

Currently, new architectural concepts are being tried out in DaimlerChrysler research cars. For example, an integration platform for the dynamic configuration and control of software modules has been developed for innovative driver assistance systems. This platform, called Agent Network System ANTS, is based on agent technology. The major advantage of this approach is the fact that the system can be flexibly extended by additional algorithms and control units and that existing software can be reused very easily. Furthermore, the architecture facilitates the cooperation and fusion of functional units which can be used for different applications in the form of a unit construction system.

DaimlerChrysler's research in the field of software architecture also comprises domain engineering and generative programming [10].

7. Conclusion and outlook

Due to the increasing use of software in the processes and products of automotive companies, software has a significant impact on product quality and company productivity. In order to meet the corresponding challenges and requirements, specific software competencies are required which cover both in-house software development as well as cooperation with suppliers. At DaimlerChrysler's Software Technology Research Lab innovative and powerful approaches are elaborated which comprise the areas of process organization, software architectures, and methods and tools. Numerous research results, such as systematic requirements engineering processes and testing techniques, could already be transferred successfully to projects, processes and development environments of DaimlerChrysler business units. Amongst other things, future work will focus on the model-based development of distributed systems, on the elaboration of a product line approach for future in-vehicle software architectures, and the integration of processes, methods and tools from the different areas of software, mechanical, and electrical engineering.

8. References

[1] E. Metzker, and M. Offergeld, "An Interdisciplinary Approach for Successfully Integrating Human-Centered Design Methods Into Development Processes Practiced by Industrial Software Development Organizations", *Proc. of 8th IFIP Conference on Engineering for Human Computer Interaction 2001 (EHCI '01)*, Toronto/Canada, Springer, pp. 21-36.

[2] M. Weber, and J. Weisbrod, "Requirements Engineering in Automotive Development - Experiences and Challenges", *IEEE Software*, January/February 2003, pp. 16-24.

[3] T. Gantner, and G. Getto, "Software Acquisition Processes - Research and Implementation at DaimlerChrysler", *Proc. of the Second World Congress for Software Quality*, Yokohama, Japan, September 2000.

[4] M. Grochtmann, and K. Grimm, "Classification Trees for Partition Testing", *Software Testing, Verification and Reliability*, Vol. 3, No. 2, pp. 63-82.

[5] www.razorcat.com/development

[6] www.hitex.de

[7] A. Rau, M. Conrad, H. Keller, I. Fey, and C. Dziobek, "Integrated Model-based Software Development and Testing with CSD and Mtest", *Proc. of International Automotive Conference*, Stuttgart, May 11-12, 2000.

[8] J. Wegener, *Evolutionärer Test des Zeitverhaltens von Realzeit-Systemen* (german), Aachen, Shaker Verlag, 2001.

[9] R. Trauter, "Software-Architekturbewertung", Tutorial (german), NetObjectDays, Erfurt, October 2000.

[10] K. Czarnecki, and U. Eisenecker, *Generative Programming: Methods, Tools, and Applications*, Addison-Wesley, 2000.