Automotive Software and Systems Engineering

Manfred Broy
Institut für Informatik, Technische Universität München, D-80290 München, Germany
broy@in.tum.de

Abstract

Information technology has become the driving force of innovation in many areas of technology and, in particular, in the form of embedded systems in vehicles. Embedded hardware/software systems control innovative functions in cars, support and assist the driver and realize new features for information and entertainment. A rapid development brought more and more digital hardware and software into a modern car only within little more than two decades. Systems and software engineering for embedded systems in vehicles is today one of the great scientific, methodological, and technical challenges. In modern cars we find all issues of software systems in a nutshell. This brings a wide spectrum of challenges for the automotive industry, including business, management, and technical issues. In the following we outline the main problems, research challenges and approaches to deal with them.

1. Introduction

More than 98 % of all programmable processors run in embedded mode today. A characteristic and technically challenging example of the extensive application of embedded digital hardware and software systems are vehicles. Today modern cars have up to 80 controllers that all have to be programmed. These processors are connected by up to five different bus systems that connect the controllers via communication protocols. The hardware system is connected to the mechanical devices of the cars and their environment via numerous sensors, actuators and network gateways. In addition, a rather complex multimedia human machine interface is available to allow the driver to access the various functions of a car.

Over the years we observe an enormous increase of digital hardware and software in cars. In the near future digital hardware and software is expected to account for up to 30% of the overall cost of a car.

Today, new innovative functions in cars are mainly realized by digital hardware and software. Up to 90 % of innovations in cars are enabled this way today.

The basic driving force of automotive hardware and software systems is, of course, Moore's law. Hardware devices quickly become cheaper and in a very cost aware industry like the car industry cost is a first class consideration. But reduced hardware costs are only an enabling factor.

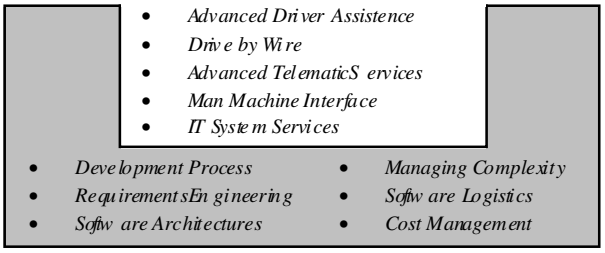

Fig. 1: Framework for Automotive Systems and Software Engineering

The possibilities for the application of automotive hardware and software systems seem nearly unlimited. There are mainly two reasons for the rapid inclusion of new software into our cars:

- new functionalities that are considered unique selling points as well as
- cheaper and better technical solutions substituting the realization of existing functionalities.

Fig. 1 outlines the software-based technical revolution in the car and the associated software engineering challenges.

A remarkable aspect is active and passive car safety. According to better and more sophisticated safety technology such as air bags and pre-crash safety functions as well as driver assistance support – all mainly realized by automotive hardware and software systems – the numbers of car accidents and also the numbers severely injured or killed people are falling continuously since years. This shows certainly better than some of the gimmick functions the value of automotive hardware and software systems.

2. The Digital Automotive Revolution

Car industry has been and is traditionally strong in the domain of mechanical engineering, of course. To some extent, however, a tradition in electrical engineering is present there, too. In the past, embedded control systems were realized mainly based on control theory.

Now within a few years digital technology and software became the driving forces due to the potentials of Moore's law that made digital technology affordable in many applications in the automotive domain.

These new options introduced a number of technical and methodological problems and challenges into the field of automotive engineering. The changes in the field can be illustrated as follows.

2.1. Digital Hardware in Cars

Modern cars of today carry more hardware and computation power than the Apollo spaceship that flew to the moon. They carry up to 80 controllers connected by up to 5 different bus systems connected to numerous sensors and actuators as well as a multimedia human machine interface and external devices such as mobile phones, personal digital assistants, and diagnostic devices for service.

2.2. Software in Cars

The software amount and the number of software based functions in cars is growing exponentially:

- Size: The amount of software in today's cars is overwhelming. In current premium cars more than 60 MB of software is installed.
- Role: Nearly all functions are controlled by software.
- Interaction: Initially the software was mainly standalone, controlling single separated functionalities. Today the functions are much more complex per se, moreover closely related and mutually dependent.
- Distribution: Today software is connected, communicates over busses and by wireless techniques also to networks outside the car.

Managing the software becomes more and more a problem.

2.3. Advanced Driver Assistance

One active field of innovations are driver assistance systems. They include functionalities such as adaptive cruise control with stop and go capabilities, lane keeping assistance, adaptive steering, stability control, brake assistance or park assistance etc. More sophisticated solutions depend on the sensors and actuators available in a car including radar and video cameras and the options to supervise and control the driver's actions and attendance.

Next steps might be context aware systems that gather information about the drivers, their habits, their conditions as well as information about the situation of the car such as street and traffic conditions. As a result the systems adapt the cars' functions to the drivers needs and traffic conditions.

2.4. Drive by Wire

One of the highly ambitious goals of the car industry are systems that realize drive by wire ideas. There are many reasons in favor of drive by wire solutions including cost issues and possibilities for new functionalities. A spectacular example is steer by wire where the steering wheel might be replaced by a joystick. Such a solution has many advantages for weight, packaging, geometry, and, in particular, for the human machine interface (HMI) in cars. New even more sophisticated functions can be immediately realized based on by wire solutions such as automatic or adaptive steering that today are realized by advanced hybrid mechatronic systems that include a mechanical fallback due to the insufficient reliability of the embedded systems of today.

Drive by wire aims at the replacement of mechanical control devices by electronic ones. Examples are brake-by-wire or steer-by-wire. First solutions are on the road for simple drive by wire functions in today's cars. More sophisticated functionalities such as steering are only realized in lab cars so far. To bring sophisticated drive by wire systems into cars has major advantages for the packaging, the costs, and advanced driver assistance. On the other hand difficult engineering problems have to be solved. The reliability of the electronic hardware/software devices is today not good enough for drive by wire.

2.5. Advanced Telematic Services

More and more cars are connected by wireless technology to systems and networks outside the car. External communication paves the road for various new functionalities such as traffic information, emergency calls, remote maintenance and diagnostics, special driver information, road pricing, fleet management and much more.

Thinking about peer-to-peer communication systems cars can be connected to cars in their neighborhoods exchanging information to support and assist their drivers and even automatically synchronize their maneuvers. Such scenarios open up to a completely new world of functions for improved safety and driver's assistance. On the other hand advanced

networking between cars introduces many interesting research issues with respect to questions of security and technical solutions.

2.6. Human Machine Interface

The man machine interface is changing in the car and becomes one of the decisive fields of innovation. The vast number of emerging functions calls for new solutions to avoid a flood of buttons and control lights. But it is not only just the user interface – the question is how the many functions can be offered to drivers in a way that is transparent and useful. Especially the smooth inclusion of driver assistance is an issue here.

First solutions are on the road bringing some of the recent HMI solutions of computers into the cars. Speech controlled functions and multimedia HMIs are found in premium cars also today. But what we see is certainly only a first step. More experience and more experiments will teach us how HMIs in cars address the needs of drivers best without any risk of drivers not paying enough attention to the traffic situation due to HMI systems that distract the drivers' attention.

More advanced questions of HMI are introduced for the sophisticated context aware, adaptive systems that may adapt the reactions of cars to changing condition, however, hopefully in a may that drivers always remain a clear understanding how to keep control.

2.7. IT System Services

The increasing hardware and software complexity needs sophisticated features for system monitoring and management. Preserving and documenting system integrity after authorized software changes have to be addressed as well as dynamic aspects like power control, load balancing, gateway functions, monitoring and logging services, etc. In addition, there are many open issues on IT security including privacy.

Next generation cars will have much more functional infrastructure that goes beyond the basic functions of operating systems or communication services. We can expect advanced sensor date fusion and integrated data management.

3. Fields of Focus in Automotive Embedded Systems

There are many challenges in automotive hardware and software engineering and in the organization of development processes for embedded systems and the system life cycle. The current systems and solutions in cars today are not sufficient to meet the expectations and needs of the automotive domain. This is explained in more detail in the following.

Reliability: So far the reliability of the hardware/software systems in the cars is not sufficient. For instance, in the domain of air traffic the calculated reliability is higher that 10^9. This means that for safety critical systems the mean time to failure is more than 10^9 hours of operation. In cars such reliability figures are not even known. For some systems reliability is estimated by a factor of 10^3 below what has been achieved in avionics. Drive-by-wire solutions in cars need a much higher reliability.

Quality: Today the quality of software in the car is not as high as required. The number of problems with hardware/software systems in cars under development, during production and in the field is enormous. For instance, the number of hardware devices that are replaced due to malfunctions in cars is much too high. Today often in the end hardware devices are replaced by the service in the case of bugs that are hard to analyze. Later analysis shows that more than 50 % of the replaced hardware devices turn out to be without bugs. Obviously hardware/software incompatibilities may be responsible for the problems. Both the number of bugs in the software is too high and the reliability of the hardware is not sufficient. In addition, the service and maintenance personnel seems to be overwhelmed by the complexity of the systems and often quite helpless when confronted with complex bugs in cars.

Supplier Management: A difficult issue is the fact that subparts of systems including software are manufactured by high number of different suppliers and have to be integrated into the final product. This brings essential problems of requirements specification, interfaces, and system integration.

Portability and Reusability: Today most of the solutions are developed and realized in a rather proprietary way for each type of car again and again. Not much of the automotive hardware and software systems is reused. A typical figure is that in a new generation car there are 10 % new functions while 90 % of the automotive hardware and software systems have been newly developed from scratch. This development overhead cannot be accepted today and even less in the future. The huge amount of software demands techniques for reuse and in particular calls for product lines.

Costs: The cost of the hardware/software systems in cars is rapidly increasing. From car generation to car generation the cost of software is growing faster and becoming more significant. The typical cost structure of the car industry oriented towards price per piece gets less appropriate. Therefore new cost models are needed to drive and manage the development of automotive hardware and software systems.

Engineering Tools: The usage of tools for requirements engineering, modeling, simulation, prototyping, code generation, and verification (testing, model checking, deductional proof) is a must. Today

the tools do not address the needs sufficiently nor do exist tool chains for a seamless development support.

4. Scientific Challenges

The fields of focus show where research issues are found for automotive embedded systems.

4.1. Requirements Engineering

Due to functional dependencies, feature interactions, new innovative functionalities and much larger design spaces in software based systems, requirements engineering becomes much more sophisticated and decisive in automotive applications. One has to find out what the users really need and expect. In addition nonfunctional requirements get more complex. After a decomposition of a system the requirements for the subsystems have to be captured and documented.

We have to deal with *functional requirements (logical requirements)*, which are properties of software related to its behavior, its offered services, and its performance, with *nonfunctional and product requirements*, which are properties of software related to its representation and documentation as well as the process of its realization and distribution, and *process requirements*, which are properties of the software development process.

4.2. Systems Architecture

So far digital systems in a car are often mainly structured in terms of their hardware architecture. With software becoming dominant, the software architecture becomes essential.

On the long run the industry needs reference hardware and software architectures in the automotive domain (see [1]). They have to relate functionality, hardware and software (see below).

4.3. Managing Complexity

The software in cars has become large and complex. There are two main sources of complexity: the sheer size of the software and the complexity of the realized functions and the growing dependency.

Size of the Individual Functions: The size and complexity of the functions is growing. Today the light control, the locking management, the antiblocking systems or airbags become more sophisticated from generation to generations of cars.

Nonfunctional Requirements: For many applications in cars there are very different requirements concerning real-time behavior, reliability or startup behavior. The idea to distinguish between critical and less critical functions is useful. Nevertheless noncritical functions are connected to critical ones and thus techniques for protection are needed. Drive by wire requires, as pointed out, a higher reliability, in particular.

Functional Connectivity and Dependency: Today the functions in cars get more and more connected. Often this happens bottom up. As a result the hidden effects take place (called “feature interactions”) which make system integration a hard task. A top down approach to the architecture in cars is needed.

4.4. Software Logistics

Not only the development of software, also its role in the production process (how to download the software into the controllers in production) and the maintenance tasks pose many tough problems. Cars will be operated in the field for a time span of 20 to 40 years. This calls for well-defined software logistics.

Software Maintenance: To maintain the software in the field includes many aspects such as version and configuration control, software update, error documentation, tracing and diagnosis.

Software Download: A critical issue is in particular the update of software in a car. Today more and more cars store their software in flash memory that can be updated by software download. To solve are basic problems of the flash time but also more sophisticate questions of the compatibility of software versions.

Software Compatibility: There are many reasons for changes in the hardware/software systems in cars. Chips may run out of production and have to be replaced by other chips. This may affect the production process as well as the maintenance. New chips may require new software. Also software may be modified during the production and operation life cycle of cars. Reasons are bug fixing or improved functionalities. In cases of local changes to the hardware or software of a system the issue of compatibility becomes crucial. How can we guarantee that the modified components of hardware or software still work together with the rest of the system?

4.5. Comprehensive Cost Management

So far the techniques for cost estimation and design to cost are highly insufficient. Although the software development and maintenance cost are now a considerable factor for a car, the costs are rapidly growing while appropriate cost models do not exist.

5. Research Issues

The challenges show where many exciting research issues are for automotive digital hardware and software. As pointed out many pressing problems need to be solved. They all have to do with an integrated

systems and software engineering that is mandatory to master the complexity and the challenges of the increased size and functionality as well as the required reliability.

There are three main lines of hope to improve the state of the art:

- Improved development processes
- Modeling techniques
- Domain specific architectures
- Tool Support

These issues are closely related and interdependent. How we structure and organize the process is closely related to the models we work out during the development. This has to be supported by tools. The main task in the development is the definition of an appropriate architecture. Again modeling techniques are needed to describe and analyze the architecture.

5.1. Process Issues

The process in automotive applications is not deeply different to other processes for defining hardware/software systems. However, it is definitely has to take into account the specific cost structure and by a highly distributed development process.

What we need are a process and a methodology that support distributed development by first, second and third tier suppliers (see [2], [4]). Therefore a proper requirements engineering where finally the architecture and all requirements for the subsystems are precisely captured and documented is a must. The quality of the work products has to be guaranteed by reviews and tests relying very much on the abilities to validate the requirements, to verify the architecture and the subsystems independently. Only this way the difficult process of putting together the results of many different teams and groups can be successful.

5.2. Modeling

In practice, many methods for modeling and specification have been suggested (SA, SADT, SSADM, SDL, OMT, ROOM, UML, ...). But none of them addresses the needs of automotive software properly. A more domain specific modeling technique is required that properly addresses the needs of the automotive domain.

5.2.1. Modeling Requirements

As requirements engineering is a key issue, the precise modeling of requirements becomes a first class prerequisite. All the modeling approaches available in practice so far do not support the modeling of requirements directly. This holds also and in particular for UML. UML's use case diagrams do not really lead to a modeling of requirements. Most of these modeling languages mainly deal with issues of architecture and implementation.

Automotive hardware and software systems are multi-functional. This means that that we have to model a large number of individual functions and also their interaction and interdependencies.

5.2.2. Modeling the Comprehensive Systems Architecture

Architectures are very significant in automotive hardware and software systems, since they are the basic guideline for breaking down the design for a distributed development and for the integration. What is needed is a comprehensive model of the system architecture of an automotive hardware and software system including the following views:

- user functionality architecture
- logical component architecture
- hardware architecture
- software architecture
- deployment of the software on the hardware architecture

All these aspects of architecture have to be captured precisely by a modeling approach.

5.2.3. Modeling the Subsystem Architecture and the Interfaces

A particular critical issue is that of the interfaces between the subsystems of automotive hardware and software systems. What we need is a modeling approach that can capture all aspects of interfaces of subsystems that allows an independent development and test of those subsystems such that we can guarantee the functioning of the overall system by

- verifying the architecture
- verifying the subsystems

This is essentially what is called a modular system construction and verification.

5.2.4. Software Architecture and Deployment

Much of the software in a car is real time. The software runs on controllers and communicates by bus systems. By the deployment it is determined which software task is carried out on which controller.

Since several software tasks may be carried out on one controller and may be communicating over a bus they have to share these hardware resources. Today, this sharing is, to a large extend, event driven. This means that it is determined by the time at which a certain event arises at what time which task is scheduled. As a consequence systems are highly nondeterministic, especially with respect to their timing. If we change the deployment or make – even just small – changes in the software of one task, the scheduling and the timing of the system may change radically. Bugs may become visible that have been remained undetected so far due to biased timing.

Moreover, it is very hard to give guarantees on time bounds.

Therefore event driven scheduling is hard to apply in highly critical application and makes it difficult to modify systems even after they have been carefully tested. Another way is static, time driven scheduling. There statically fixed time slots are reserved for the tasks running on the controllers and also on the bus systems. The system becomes more predictable. Nevertheless, resources may be wasted if tasks do not fully exploit their time slots. Time driven scheduling only gradually finds its way into automotive hardware and software systems. A comprehensive methodology for time driven solutions is still missing, however.

5.3. Reliability and Quality

To achieve the reliability that is required for innovative software solutions we have to extend and transfer results from formal methods in a pragmatic way to practice. So far the reliability is not sufficient for automotive hardware and software systems as we have already pointed out.

Due to the rising complexity and combinatorial explosion a pure system level test by system in the loop as it is mainly done today is no longer sufficient. Even today it cannot guarantee a sufficient reliability. What is needed is process ensuring comprehensive quality and reliability based on advanced modeling techniques. A promising approach here is model-based testing (see [3], [14]).

Sophisticated error management is required to improve the quality and reliability. This is again an architectural issue. This way questions of error detection, error logging, error recovery and fail safe can be tackled.

5.4. Product Line Engineering

Cost, quality, reliability can be improved on the long run in cars only by an advanced product line engineering for automotive embedded systems. All what we have described, such as an improved optimized standardized development process, domain-specific system architecture and modeling methods for it and a well-understood quality process are the prerequisite for a product line engineering of embedded system in cars (see [5]).

Over the years with a better and better maturity of the automotive industry we will arrive at such a point where automotive systems in cars show uniform architectures and a modularity that solutions can be reused and adapted many times, such that 10 % new functionality does not need much more rework than additional 10 % development.

ACKNOWLEGEMENTS

Its is a pleasure to acknowledge helpful remarks Jan Romberg and from colleagues from BMW CarIT, in particular Alexandre Saad and Ulrich Weinmann.

References

[1] AUTOSAR - Automotive Open System Architecture, http://www.autosar.org

[2] J. Bortolazzi: Prozesse und Methoden für die Entwicklung von Kfz-Software bei der DaimlerChrysler AG. In Informatik 2004 - Informatik verbindet, Band 2, Beiträge der 34. Jahrestagung der Gesellschaft für Informatik e.V., 2004

[3] M. Broy, B. Jonsson, J.-P. Katoen, M. Leucker, A. Pretschner (Hrsg.): Model-Based Testing—a tutorial volume. Springer LNCS 3472, 2005

[4] M. Broy, A. Rausch: Das neue V-Modell XT – Ein anpassbares Vorgehensmodell für Software und System Engineering. Informatik Spectrum, Juni 2005

[5] P. Clements, L.M. Northrop: Software Product Lines – Practices and Patterns, Addison-Wesley, 2001

[6] R. Kneuper: CMMI - Verbesserung von Softwareprozessen mit Capability Maturity Model Integration. dpunkt.verlag, November 2002

[7] Frischkorn, H.-G.: Offene Systeme als Basis der Systemintegration der Zukunft. 8. Internationaler Fachkonkress Automobil-Elektronik, Ludwigsburg, 2004

[8] H. Gentner, M. Ehrmann, C. Salzmann: Model Based Development of Automotive Human Machine Interfaces. Convergence, SAE, Detroit, Michigan, October 2004

[9] B. Hindel, K. Hörmann, M. Müller, J. Schmied: Basiswissen Software-Projektmanagement - Aus- und Weiterbildung zum Certified Project Manager nach dem iSQI-Standard. iSQI-Reihe, dpunkt.verlag

[10] M. A. Jackson: Software Requirements & Specifications — a lexicon of practice, principles and prejudices, Addison-Wesley Press, 1995

[11] P. Liggesmeyer "Software-Qualität", Spektrum Verlag 2002

[12] N. G. Leveson: Safeware – System Safety and Computers, Addison-Wesley, 1995

[13] D. A. Peled: Software Reliability. Springer, 2001

[14] A. Pretschner et al.: One Evaluation of Model-Based Testing and its Automation. Proc. ICSE 2005

[15] Th. Scharnhorst: Systementwurf für Elektronikarchitekturen im Fahrzeug. 4. Braunschweiger Symposium Automatisierungs- und Assistenzsysteme für Transportmittel. VDI Reihe 12, 525, 2003

[16] Schäuffele, J.; Zurawka, T.: Automotive Software Engineering – Grundlagen, Prozesse, Methoden und Werkzeuge. Vieweg Verlag, 1. Auflage, 2003

[17] I. Sommerville, Software Engineering, Addison-Wesley, 2001

[18] A. Spillner, T. Linz Basiswissen Softwaretest: Aus- und Weiterbildung zum Certified-Tester, dpunkt-Verlag, 2003

[19] ISO 9126: Information technology – Software product evaluation – Quality characteristics and guidelines for their use, Genève, ISO/IEC, 1991.

[20] Motor Industry Software Reliability Association: Report 5 – Software Metrics, Motor Industry Research Association, Februar 1995.