The Software Car: Building ICT Architectures for Future Electric Vehicles

Christian Buckl, Alexander Camek,
Gerd Kainz, Carsten Simon
ForTISS GmbH
Cyber-Physical Systems
Munich, Germany
Email:{buckl,camek,kainz,simon}@fortiss.org

Ljubo Mercep, Hauke Stähle, Alois Knoll
Technical University Munich
Robotics and Embedded Systems
Munich, Germany
Email:{mercep,staehle,knoll}@in.tum.de

***Abstract*—Disruptive technologies have the potential to change markets dramatically. The switch from internal combustion engines to electrical engines is such a change. But electric engines for vehicles are only the catalyst for the real change. Most significantly, the architecture and role of information and communication technology (ICT) will change for the vehicle of the future. This paper discusses the results of a study conducted in Germany on the role of ICT architectures. Furthermore, it will present an experimental platform that implements the vision of this study.**

I. Introduction

Information and communication technology (ICT), in the form of electrics, electronics, and software in vehicles, is already essential for the competitiveness in the automotive industry. Its most notable effects are the improvement of driving performance and comfort, and the enhancement of both passive and active safety. But the effects go further in the case of electric vehicles: ICT becomes the backbone of all relevant functions. For that reason, architectures and technologies for vehicle ICT cannot be viewed merely as a frame for gradual evolutionary innovations as before. This approach would result in a rise of complexity, which becomes too hard to manage. Instead, the ICT architectures must be revised so farsightedly that they can perform their indispensable role in future cars. The disruptive technology change currently taking place might be the chance for such a revision, but the need for a new ICT architecture is already apparent.

This paper discusses the results of a study [1] conducted in Germany[1]. The goal of the study was to investigate the potential changes of the ICT architectures induced by the shift to electric vehicles. It is based on 240 interviews worldwide with experts from all relevant fields including original equipment manufacturers (OEMs), original equipment suppliers (OESes), politics, and consumer organizations. In addition, desk research was used to analyze the state-of-the-art in industry and research both regarding architectures and technologies.

This work is supported by the German Ministry of Economics and Technology with grants Nr. 01ME12009 and Nr. 01ME10003.

[1]The whole report is available in German at www.fortiss.org/ikt2030

Section II summarizes the ICT architecture in today's vehicles. Based on the societal and technological trends described in Section III, the corner stones of future ICT architectures are discussed in Section IV. We suggested to build evaluation prototypes which implement such an architecture. Section V presents the first version of this architecture. The paper is summarized by a conclusion and a description of future work in Section VI.

II. ICT Architecture in Today's Vehicles

Over the past 30 years, ICT - in the form of vehicle on-board electronics together with the associated software - has made significant innovations in the automotive domain possible: from the anti-lock braking system in 1978, over electronic stability control in 1995 to the emergency brake assist in 2010. According to current estimations, ICT contributes 30 to 40% to the total value added in automotive construction. At the same time, ICT architecture has also become more complex: because of the technologies employed, in terms of the functions performed, and in regard to the supply chain. ICT and especially its software, has expanded significantly, from about 100 lines of code (LOC) in the 1970s to as much as ten millions LOC [2]. Software is now one of the major driving factors behind the automotive innovations. Today, 80% of innovations in cars are a direct product of the technology transfer from the domain of computer systems [3].

Nevertheless, the use of ICT lags well behind the technical possibilities. Safety is achieved mainly by passive safety measures. Proactive safety functions (such as emergency brake assist) that make heavy demands on ICT are treated with great wariness. One reason is the complexity that is introduced by these functions, as they are highly interconnected. More advanced driver assistance systems which integrate and fuse a large amount of sensor data require access to sensors, actuators, and the human machine interface (HMI). In other areas that offer great potential to help products stand apart from the crowd, such as infotainment or telematics, ICT architecture in cars has not been keeping pace with developments in other sectors. Holistic approaches for handling all the major activities inside the vehicle are held back by the legacy concepts. This results in vehicles which are already outdated in

the moment of purchase. For example, between design freeze and start of production of a car, several cell phone generations pass [1].

The major reason for the slow progress is the evolutionary development of architecture, avoiding dramatic changes for as long as possible. This approach gave rise to immense complexity of system interconnect which is getting harder to handle. Today, German premium vehicles have 70 to 100 electrical control units (ECUs) [3]. They are networked by a way of a highly complex wiring harness using multiple bus systems, for example, for the engine chamber, chassis, passenger compartment and infotainment, as well as different communication technologies, including LIN, CAN, FlexRay, and MOST. The result is that in today's vehicles this complex ICT architecture increasingly becomes a barrier to innovation. Due to the high complexity of the network and the fragmented development process with plenty of Tier-One-Suppliers, the current ICT structure becomes a likely source of faults leading to an increase of warranty costs. McKinsey estimates that OEMs could achieve 15 to 20% more earnings before interest and taxes (EBIT) if no warranty costs would be incurred by software faults [4].

III. Societal and Technological Trends

One of the results of the study is that several complex functions must be integrated in the vehicle until 2030 due to societal trends. In the following the main societal trends are discussed and examples for relevant functions are given:

1) *Energy and Cost Efficiency:* ICT can help to reduce the required energy per kilometer by two main factors. Firstly, it can implement functions in software previously realized mechanically or in the form of discrete hardware units to reduce the weight of the car (e.g., steer or brake-by-wire). Secondly, intelligent predictive management functions can reduce the energy consumption.
2) *Zero Accidents [5]:* By implementing pro-active safety functions, ICT can help to reduce the number of accidents. Future vehicles must be able to perceive and interpret their environment in order to act (semi-) autonomously in dangerous situations[2]. This trend is strengthened by the aging society.
3) *Seamless Connectivity:* Future passengers will demand for seamless connectivity including the state-of-the-art in infotainment. It must be possible to update the vehicle frequently to keep track with the advances in multimedia and infotainment technology.
4) *Personalization:* The consumers will want to retain a certain level of personalization in spite of the growing car-sharing initiatives. One approach would be transferring secondary vehicle functions to a personal mobility device. This goes in line with the aforementioned demand for seamless connectivity.

To implement the demands that arise by the societal trends, the following challenges have to be considered:

1) *Function Amount:* The amount of functions in a car tends to grow from generation to generation. This leads to an increase of complexity and additional requirements of hardware resources.
2) *Communication Demand:* With an increasing performance of the individual functions and a stronger mutual dependency, the amount of logical connections and exchanged data increases. While automotive functions used to be separated from each other, the future car will have strongly interwoven systems in order to enable the realization of highly-sophisticated functions. Especially advanced driver assistance systems need the state of different sensors in order to build an usable abstraction of the environment. Furthermore, the increasing quality of media processed in the infotainment systems contributes to the rising data exchange.
3) *Functional Segmentation:* The classic segmentation of the functions into the domains like chassis, infotainment etc. does not reflect the flow of processing anymore. As a consequence, the integration of new domain-crossing functions is complex and cost-intensive. Those cross-links increase the probability of design mistakes and thus push the costs for testing enormously.
4) *Adaptability and Flexibility:* To achieve the goal of higher personalization, the car has to be adaptable to the driver's needs. This does not only include the customization of the HMI, but goes far beyond. In the future, customers will not accept vehicles that are not equipped with the latest safety functions. Hence, new functions, but also sensors must be integrated into the vehicle easily.
5) *Security:* New communication channels always open the door for intruders. As the future electric vehicle will exchange data with other cars as well as keep a connection to the Internet, special consideration has to be taken in order to eliminate the risk of unauthorized access to the vehicle subsystems. Various mechanisms used for personalization and adding of new functionality open doors for new kinds of attacks on the system.

The societal requirements are backed up by the technological trends. Nearly all technology required to implement solutions for above mentioned challenges is available. Other domains, such as the avionic industry have shown that by introducing new architectures the above mentioned challenges can be solved. The major technological trends that help during this transformation are:

1) *Encapsulation and Modularization:* There is a trend towards greater miniaturization of hardware and development of intelligent modules. Highly integrated mechatronic components are evolving, such as smart sensors or actuators. These intelligent components encapsulate functionality by executing tasks usually found on higher architectural levels and local control loops. This helps to reduce the complexity of the overall ICT architecture.
2) *Standardization:* Other domains have shown that by

[2]Activity in Europe done by ERTICO http://www.ertico.com/

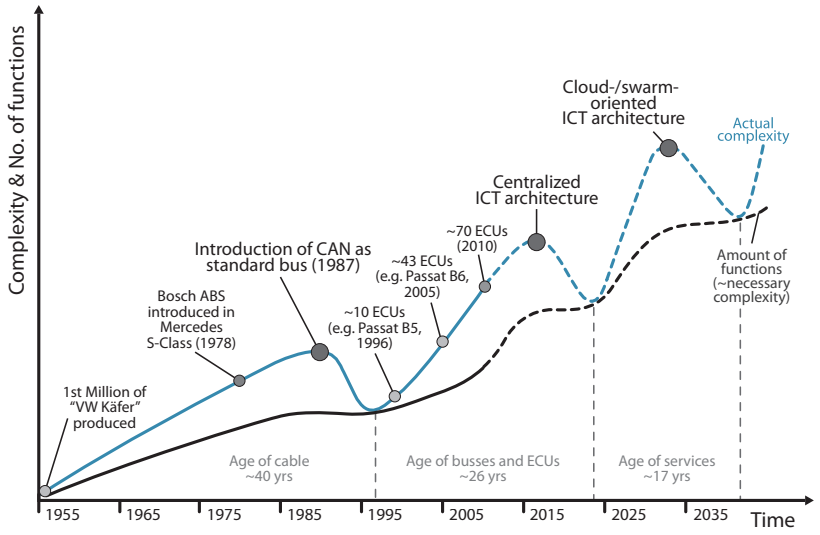

Fig. 1. Evolution of complexity in ICT architectures

providing standardized and high-level interfaces, the complexity can be mitigated. Relevant approaches can be taken from the avionics, robotics, and PC industry domain.

3) *Virtualization:* The complete underlying ICT platform is further virtualized in the form of middleware. This abstraction layer takes the responsibility for the realization of non-functional requirements, such as fault-tolerance. Using this approach, functions can be distributed even outside the vehicle. The reasons could be insufficient processing power and increase in energy efficiency, amongst others. The automotive industry is already thinking in this directions, as the effort regarding AUTOSAR [6] shows. However, the middleware must provide much more functionality than the functionality that is already today included in AUTOSAR. Details on possible functionality provided by the middleware are given in the next section.

IV. Future ICT Architecture

In the evolutionary development of vehicle architectures, shown in figure 1, there is an evident trend for architectures to become more complex than required for the according gain in functionality. This phenomenon of evolutionarily grown systems was already described by Brooks in 1987 [7]. The results are larger integration costs and a decreasing innovation curve. Only a substantial revision of the architecture and a technology leap can bring the actual complexity down to the necessary complexity. In substance, the level of abstraction at which new functions are integrated must rise. For that purpose, part of the platform must be virtualized. This platform will become a standard component - a commodity - and the complexity and price of that platform will shrink.

[3]For example GM OnStar http://www.onstar.com

[4]see GM OnStar or driveNOW https://www.drive-now.com/

This process has already been observed in the automotive industry in the past. To reduce emissions and improve comfort in the 1980s, a wider usage of microcontrollers became necessary. Because it was almost impossible to cable all these electronic modules together, complexity relatively quickly became a big problem. A solution came in form of communication busses like the CAN bus. CAN virtualized the physical connection - in this case, the cable. It thus became significantly easier to introduce new functions, because integration no longer had to be done by changing the wiring, but by integrating at signal level.

Today's ICT architecture again faces similar problems, but this time because of the large number of ECUs. The required integration and testing effort to exclude undesired interactions of features or functions is growing enormously. The study therefore suggests the following main design principles for future ICT architectures, which are influenced by [8]. The design principles are grouped by their relation to hardware and software.

For the hardware the study suggests the following design principles:

1) *Centralized Computer Architecture:* Future ICT architectures should be based on centralized and scalable computing units, which execute all hardware-independent functions. Scalable hardware solutions are desired to enable the usage of standardized components for all types of vehicles produced by a company. Such a standardized hardware reduces costs for maintainability and unit costs due to large-scale production. The study assumes that the individual modules will be high-performance multicore systems.
2) *Highly-integrated Mechatronic Components:* Sensors and actuators will become smart components that communicate over standardized data interfaces with the centralized computers. The components will already provide

	ICT of today's vehicles	ICT of future vehicles
Energy- / Cost-Efficiency	• Optimization at component and/or domain level	• Optimization at system level • Weight reduction • Intelligent predictive control manager
Zero Accidents	• Pro-active safety functions are slowly getting integrated • Safety relies mostly on passive mechanisms	• Pro-active safety functions
Seamless connectivity	• Integration of CE devices possible • Interaction with cloud possible[3] • No software update without proprietary firmware or a visit of the local workshop	• Integration and interaction with CE devices • Interaction with cloud possible • Software updates over the air (OTA)
Personalization	• Currently possible but vendor specific[4]	• Vendor independent

TABLE I
COMPARISON OF ICT ARCHITECTURES IN TODAY'S AND FUTURE VEHICLES

internal functions to preprocess the raw sensor data. However in contrast to today's approaches, the preprocessing will be reconfigurable so that the component can adapt to new system requirements. The integration on the mechanical level is reduced to connecting to the communication backbone and to the power supply network. The real integration takes place at the software level.

3) *Standardized Communication Backbone:* All interviewed experts assumed that in the future a real-time Ethernet backbone will replace current communication buses. It is important to note that additional local communication protocols might be used for the interaction with sensor or actuator components. However, due to abstraction, these protocols do not have to be taken into account at system level.

In addition, the study suggests the following software design principles:

1) *Consideration of Extra-functional Properties:* Operating system and middleware technology also needs to take the extra-functional properties into account. In contrast to standards like AUTOSAR[6], that focus predominantly on the functional integration[5], the future middleware technology must support guarantees with respect to extra-functional properties such as timing, fault tolerance, and security. By providing basic functionality regarding these aspects, the developers can focus on the application-specific requirements. This leads to a significantly reduced complexity of the development process.
2) *Plug&Play Capabilities:* New functions, sensors, actuators, and other components can be easily integrated in the vehicle if the platform supports Plug&Play. At the hardware level, standardized plugs and sockets are needed to allow an easy integration of new components. At the software level, protocols must be defined to support the detection of new components and the exchange of all information required to integrate a component into the system. The current ICT architecture requires a manual reconfiguration of the whole system and especially the communication flow after the addition of components. Future Plug&Play concepts will enable changing and adding of components with minimal human intervention. The Plug&Play concept will allow customers to change and personalize the vehicle. Furthermore, these changes will not be limited to changes of the infotainment domain, but of all other domains as well.
3) *Resource Awareness:* A Plug&Play mechanism needs information about assigned and available resources to be fully functional. A dedicated component should manage resources to execute safety critical and non-safety critical functions without interference and to satisfy the extra-functional properties of the individual components. It should also change the resource assignments to enable the integration of new components. Therefore, all resource requirements of the components must be known at run-time. Based on this knowledge, the configuration can be adapted if the application has changed. Reconfiguration can be triggered by the addition or removal of application components, the addition or removal of resources or component failures. In addition, this mechanism can be used to improve the distribution of work load.
4) *Abstraction of Communication Infrastructure:* In current ICT architectures, every communicating component has to know what kind of infrastructure is used in order to fulfill timing or quality-of-service (QoS) requirements. To decrease the engineering complexity in the future,

[5]AUTOSAR 4.0 recently introduced mechanisms to describe the timing and also fault-tolerance requirements. However these techniques must be significantly further improved.

the communication infrastructure must be abstracted. Instead, a suitable interface must be provided that allows the specification of QoS requirements for a specific communication.

5) *Energy Management:* The cruising range of modern electric vehicles is hampered by the energy requirements of rapid acceleration and the usage of comfort systems. Current state of energy storage technology requires a prudent approach in order to provide acceptable levels of mobility and driver satisfaction. Therefore, the platform must provide functionality for a basic energy management. The according software modules monitor and coordinate the needs and transformation of energy as well as the adequate configuration for each component. It is also necessary to properly react to overloads as well as insufficient energy levels. Further, predictive functions for energy management can be implemented at the application layer.
6) *Sensor Fusion:* Input given by a variety of potentially faulty sensors has to be combined to generate complex data like the vehicle status. Multiple sensor flows can be combined to yield additional data, reduce the strain on the processing part of the system or increase data quality. Another important task is to distribute the generated data to relevant functional components. Generic functions for sensor fusion must be provided by the software infrastructure.
7) *Functional Safety:* The ICT platform must provide basic functionality for fault recognition and fault tolerance, such as redundancy management and the control of fault propagation. The main concept is the support of applications with mixed criticality via spatial and temporal partitioning of processes.
8) *Security:* In order to protect internal vehicle data and individual services, security mechanisms have to be implemented at the platform level. This comprises mechanisms for authentication, authorization, key management and access policies. The platform might, for example, refuse the access of a third party application to GPS data in order to prevent tracking and recording the movement of the vehicle.

The above mentioned architecture is of course very visionary. However, already today one can see a trend in this direction. As figure 2 shows, ICT architectures could thus develop in three steps. In the first step, which is already happening today, ICT modules are integrated and encapsulated at a high level. In the second step, the ICT architecture could be reorganized with respect to all functions relevant to the vehicle. And finally, a middleware that integrates both the functions relevant to driving and the non-safety-critical functions for comfort and entertainment would make it possible to customize vehicles for their drivers by way of third-party software.

Fig. 3. Design of the eCar evaluation platform

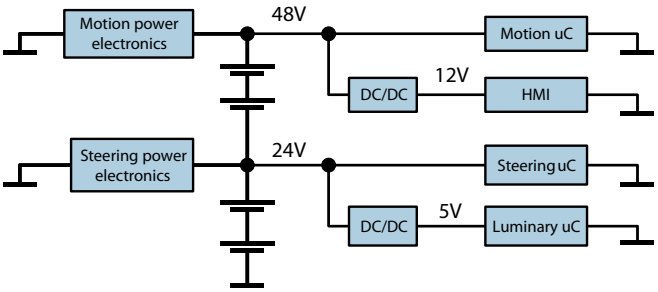

Fig. 4. Overview of the energy network of the eCar

V. Prototypical Implementation of Suggested ICT Architecture

Based on the results of the study, we have built an experimental platform to implement the suggested ICT architecture. Figure 3 shows the eCar evaluation platform, videos are available online[6]. The most challenging hardware setup has been chosen to show the capabilities of future ICT architectures: an electric car with four independently controllable driving/steering units interconnected by a modified Ethernet-based network. The driving and steering actuators are combined to eCorners following a concept developed by Siemens VDO [9]. The eCar is composed of four such eCorner modules with a weight of around 50 kg each. As the eCar does not have any mechanical brakes, braking is realized via in-wheel driving motors. Recuperation is used to regain a part of the kinetic energy during deceleration. No mechanical axles are used for the synchronization of the modules. Instead, the whole control is based on a distributed X-by-wire system. No mechanical fallback solutions are integrated in this vehicle, so that the whole reliability and safety has to be ensured by software.

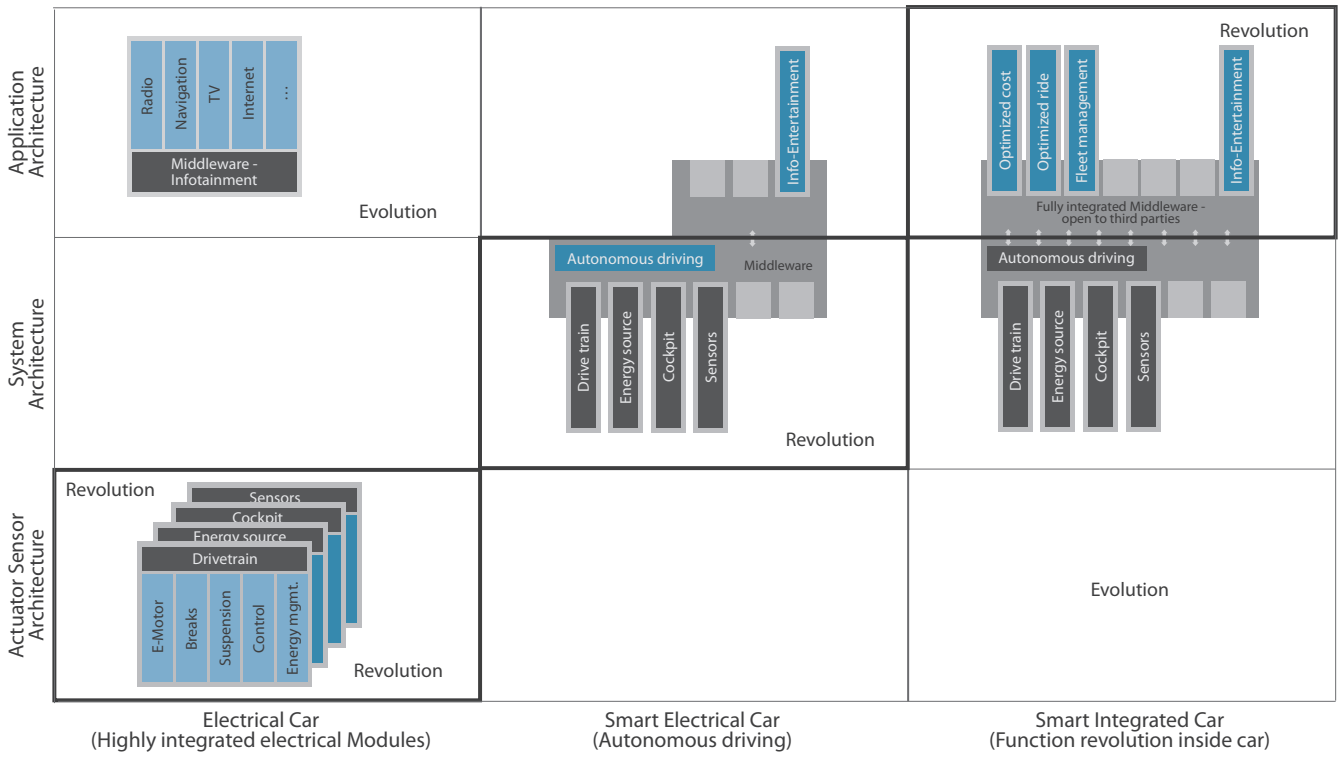

Fig. 2. Possible scenarios for the further development of ICT architectures

A. Electromechanical Setup

The complete drive power of the eCar is 8 kW. Each of the in-wheel engines has 2 kW and a maximum torque of 160 Nm. As presented in figure 4., the energy is provided by four lead-acid batteries. The batteries are used to supply 48 V for the in-wheel driving motors, 24 V for the steering actuators and 48 V, 24 V, 12 V, and 5 V for the ECUs. 12 V and 5 V are provided via down-step DC/DC converters. The eCar is controlled with a sidestick connected to the HMI unit. The current state of the evaluation platform is presented on a 10-inch touchscreen, which can also be used as an input device, i.e., to change between different driving modes. The outline of the eCar is approximately 2.25 x 1.25 x 1.75 m (L x W x H) and its weight is about 600 kg. The eCar is constructed to carry one passenger up to a maximum speed of 50 km/h.

Due to the fact that all eCorner modules can be controlled independently, it is possible to realize different driving modes:

- Normal Mode: conventional two-wheel steering of the front wheels, of the back wheels, or four-wheel steering
- Vector Mode: all four wheels rotate while being parallel to each other
- Parking Mode: used to drive sideways into a parking lot
- Turn-on-place Mode: rotating around the center of the vehicle
- Parking brake Mode: fixes the eCar on place without mechanical brakes

[6]http://ecar.fortiss.org

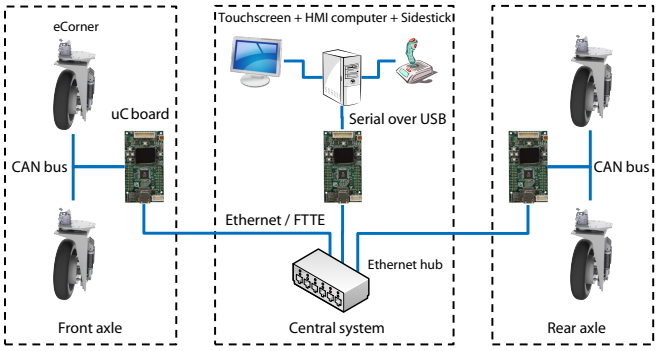

Fig. 5. Current architecture of the eCar

The modular and open construction of the evaluation platform makes it possible to easily attach or detach sensors and ECUs as needed. In addition, different network topologies can be integrated into the eCar. Hence, it is suitable for the evaluation of different ICT architectures.

B. ICT Architecture

The current version of the eCar accepts driver's input via a sidestick and touchscreen. The architecture is shown in figure 5. It consists of distributed, heterogeneous, and interconnected hardware components. The basic components are Stellaris LM3S8962 Evaluation Kit boards with an ARM Cortex M3

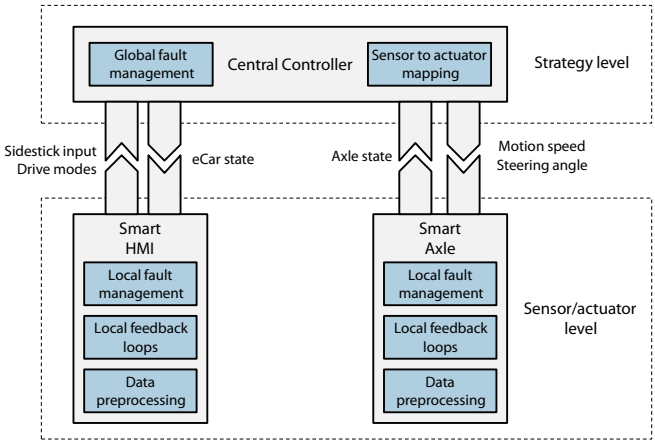

Fig. 6. Software architecture of the eCar

CPU operating at a clock rate of 50 MHz. FreeRTOS[7] is used as real-time operating system on the Stellaris boards. For communication, an own real-time Ethernet protocol, the Flexible Time-Triggered Ethernet (FTTE) protocol, based on the standard IEEE 1588 [10], was developed. It supports, similar to FlexRay, time- and event-triggered communication, but can be extended to allow dynamic slot allocation at run-time for the time-triggered phase. Furthermore, it operates at a higher speed (currently 100 Mb/s).

Two controllers, one at the rear and one at the front axle, are used to implement the smart sensor/actuator layer with respect to the eCorners. Since no power electronics with Ethernet interface were available, a CAN bus is used to control the individual actuators. The strategy layer is implemented on an additional Stellaris controller. A fourth controller based on an Intel Atom Dual Core CPU is used to implement the smart sensor and actuator layer with respect to the HMI.

The experimental platform is currently able to successfully execute the driver input. The driver can select between four-wheel steering, vector, parking, turn on place, hand brake and emergency brake mode. According to the selected drive mode, the user input is handled in different ways. For example, the mode turn on place only reacts to side movements of the sidestick and translates them into the related direction and speed of the rotation.

C. Software Architecture

Although the hardware setup is distributed, the software architecture shown in 6 follows the proposed centralized approach. We distinguish a strategy level, implemented on the central controller, and a smart actuator or sensor level, represented by the front controller, rear controller, and the HMI unit.

The front and rear axle can each be seen as a smart sensor/actuator consisting of two eCorners and an axle controller. These axle controllers receive abstract commands from the central control unit and are responsible on their own for a correct execution of the received commands. For that purpose, local control loops are implemented on the axle controllers guaranteeing a fast response time. Abstract status and fault reports are sent back to the central control unit in order to allow an observation and adaption of the strategy in the case of faults.

The HMI unit is treated as a smart component in our prototype. It is responsible for a correct measurement of the user input including data filtering and checking and for displaying the state of the vehicle. Independent of the selected driving mode, the state of the sidestick is sent to the central controller for further processing. As the input of the sidestick is not interpreted in connection to the eCar state, it is easily possible to exchange the whole HMI system with a more advanced component, that is for capturing the driver's wish of motion.

Finally, the strategy level is implemented on the central controller. It is responsible for the mapping between the sidestick input and the state of the eCorners. It also interprets the feedback sent from the axle controllers and reacts on faulty system states. It is very noticeable that the driving functionality can be easily adapted by only changing the software running on the central controller, the sensor/actuator layer does not have to be altered in that case.

VI. Conclusion

Electromobility will make information and communication technology (ICT) in vehicles much more important. This will have far-reaching effects on the automotive industry. This paper discussed the potential development of the ICT architecture with respect to the predicted shift towards electric vehicles. In current vehicles, the complex ICT architecture is increasingly becoming a barrier for innovations. It is obvious that complex functions can only be implemented with reasonable costs in the future if the architecture is revised drastically. A revised architecture might also lead to faster development times and drastic cost reductions for today's functionality. Development times and costs are reduced due to the usage of standardized platforms and reuse of existing components. In the context of the rise of electric vehicles, there exists the danger that a new market player will introduce such an architecture due to the lower barriers to entry. Current OEMs with the associated supply chains might be reluctant to adapt, which might result in large market losses in the long term.

The paper also discussed the main characteristics of a new architecture based on the outcomes of a study taking into account 240 interviews world-wide. This architecture will consist of centralized scalable computer units, a unified communication backbone, smart sensors, and smart actuators.

A prototype was developed in order to evaluate the suggested concepts. Due to its four individually controllable wheels, the prototype poses interesting challenges on the ICT architecture. The current setup demonstrates not only that these challenges can be met, but also that all the necessary functionality can easily be built upon a centralized architecture.

[7] http://www.freertos.org/

In the future, the main focus of our research will be on the middleware. The goal is to develop an open source middleware as a prototypical implementation for future usage in the automotive domain. This middleware should provide the following mechanisms:

1) *Built-in Mechanisms for Fault-tolerance:* The middleware should support the execution of application components with different criticality levels (mixed criticality) on centralized controllers. Furthermore, built in mechanisms which make use of redundancy in order to achieve fault-tolerance for highly critical functions should be provided.
2) *Support of Plug&Play:* Future automotive ICT architectures must support easy addition and update of software and hardware components. Therefore, we will implement mechanisms in the middleware to easily integrate these components without compromising the safety and security of the system. Key concepts are standardized hardware and software interfaces and the definition of service discovery mechanisms based on a data-centric communication approach.
3) *Automotive-specific Functions:* The middleware should also provide functions that are not included in existing middleware technologies today, but are located at the application level. Examples are basic energy management or data fusion mechanisms.

Furthermore, we will implement demanding high-level applications to demonstrate the ability of the ICT architecture to satisfy the future requirements. Autonomous driving and highly integrated energy management are examples for such complex functionalities with high demands on the underlying architecture.

ACKNOWLEDGMENTS

The authors would like to thank the group Embedded Systems and Robotics at Technical University Munich for their support during the mechanical construction of the platform.

REFERENCES

[1] "The software car: Information and communication technology (ict) as an engine for the electromobility of the future," ForTISS GmbH, Tech. Rep., March 2011, summary of results of the "eCar ICT System Architecture for Electromobility" research project sponsored by the Federal Ministry of Economics and Technology.

[2] R. N. Charette, "This Car Runs on Code," *IEEE Spectrum*, vol. Green Tech, Advanced Cars, Feb 2009. [Online]. Available: http://spectrum.ieee.org/green-tech/advanced-cars/this-car-runs-on-code/0

[3] M. Broy, I. Kruger, A. Pretschner, and C. Salzmann, "Engineering automotive software," *Proceedings of the IEEE*, vol. 95, no. 2, pp. 356–373, Feb 2007.

[4] U. N. D. J. Hoch, W. Huhn and A. E. Zielke, "The race to master automotive embedded systems development," *McKinsey Brochures*, 2006.

[5] "TOWARDS ZERO - ambitious road safety targets and the safe system approach," OECD, Tech. Rep., 2008.

[6] *AUTomotive Open System ARchitecture (AUTOSAR) Release 4.0*, AUTOSAR Std.

[7] F. P. Brooks, Jr., "No silver bullet essence and accidents of software engineering," *Computer*, vol. 20, pp. 10–19, April 1987. [Online]. Available: http://dx.doi.org/10.1109/MC.1987.1663532

[8] L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice*, 2nd ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2003.

[9] "Siemens VDO Making a Case for In-Wheel Systems: the eCorner Project," *Green Car Congress*, Sep 2006. [Online]. Available: http://www.greencarcongress.com/2006/09/siemens_vdo_mak.html

[10] "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems," *IEEE Std 1588-2008 (Revision of IEEE Std 1588-2002)*, pp. 1–269, 2008.