The vehicle architecture of automated driving level 2/3
The requirements for automotive system design and software development are increasing due to the three most important future issues of the automotive industry: autonomous driving, software updates over the air and drive system electrification.

By Rudolf Grave

The current E/E (electric/electronic) architecture in the vehicle integrates one or a few vehicle functions per control unit. This increases both the number of control units and distributed software functions and the complexity of connectivity respectively. In this context, the E/E architecture must perform an increasing number of driver assistance functions. Estimates of software complexity assume that the more than 100 control units in a current premium vehicle contain more than 100 million lines of code.

At present, single or closely related functions are each implemented on a separate control unit. The availability of higher-performance systems on a chip (SoC, e.g., Renesas’ R-Car H3, NXP BlueBox or NVIDIA DRIVE PX) suitable for automotive applications and the necessity to save weight, for example, by reducing control units or cabling result in a desire: to integrate multiple functions on a domain controller (responsible, for example, for the body, chassis or engine) or even less central computers.

This paradigm shift changes the vehicles' E/E architecture considerably. It involves the introduction of service-oriented communication and dynamic operating systems, which, in turn, must meet the requirements for real time, functional safety and security. Moreover, the use of dynamic control units allows adding functions that are not available when the vehicle is launched.

The following article deals with the different technologies for future E/E architectures and shows changes, risks and new opportunities.

**Figure 1** shows the probable future E/E architecture. At the heart are one or few central computers that communicate via a vehicle-internal Ethernet backbone. The key element of the vehicle is the gateway: It separates the user interface domain (infotainment system/smartphone connection) from the drive domain (drive system, brakes, battery management) and connects the vehicle to the OEM's back-end system using a so-called smart antenna. The main task of the smart antenna and the gateway is to implement different security layers such as firewalls and intrusion detection. Furthermore, the architecture will use mechanisms for secure on-board communication between the control units.

The connection to the back-end system enables many new functions. For example, the vehicle can be provided with environmental data such as road conditions, free parking spaces or the latest offers from the vehicle manufacturer. These online services and the option to enable functions (e.g., driver assistance systems) give vehicle manufacturers the opportunity to generate revenue beyond the car's time of sale, i.e., even when it is in use.

The permanent online connection to the vehicle allows the OEM to collect user data and thus obtain more information about the reliability and wear of the components used. Sources of error in hardware and software and associated environmental data can be detected via the diagnostic interface, the software can be improved at the manufacturer and an update can be promptly downloaded to the car - similar to the smartphone app updates that users have grown used to over the years.
Autonomous or highly automated driving requires that the vehicle be aware of its surroundings. The environmental model is built by the so-called sensor fusion. It combines the camera, radar, LiDAR (Light Detection And Ranging), and ultrasound data into a single model. These different sensor technologies are required as some of the systems have weaknesses that can be compensated for by other technologies. For example, unlike radar, camera systems may not be able to detect bright objects when blinded by the sun.

In the future, these complex calculations will be made by the central computer in the vehicle. The processors used will be heterogeneous multi-core processors, presumably with several cores, GPUs, and Gbit Ethernet channels. For safety-critical functions such as plausibility checks, monitoring and result validation, additional safety cores will be integrated onto the chip or a second processor will be integrated onto the board. With ARM Cortex A50/A57, Renesas’ R-Car H3, Cortex R7 and Infineon Aurix, such systems already exist.

In contrast to these complex multi-core systems, many control units were 16-bit single-core systems just ten years ago. For the suppliers, this leap in technology means the demanding task of building competence in the field of software. Software used to be only a cost factor that was part of a brake, including a control unit. In the future, the software function will be the real value. This will disrupt the existing supply chain and enable new business models. Who will benefit from this development? Probably manufacturers who early use integrated toolchains for system design, time modeling, code generation and verification as well as validation to manage the increasing software complexity and therefore the costs.

At present, most control units use statically configured operating systems implemented according to the AUTOSAR or OSEK standard. During configuration time, these systems define scheduling and resource utilization that can only be changed to a limited extent during run-time. The static configuration has the advantage that it is easier to verify that a function is executed within a certain time span. In the case of the side airbag, for example, a decision has to be made in just a few milliseconds and the airbag has to be deployed.
For less time-critical systems with more complex multi-core processors and different external interactors (software update, user input), dynamic operating systems have their strengths. The most important application scenarios are:

- Support of reconfiguration during run-time
- Service-oriented services and communication
- Partial software updates
- Simplification of software development by using POSIX interfaces instead of statically generated XML-based interface descriptions

In this context, the AUTOSAR consortium introduced Adaptive AUTOSAR (see Figure 2). It includes a POSIX OS, that runs either directly on a multi-core processor or in a hypervisor environment if multiple operating systems are to be integrated in parallel. The Adaptive AUTOSAR working groups of different OEMs and suppliers define the special services for use in automotive applications such as diagnostic services, security services and SOME/IP. The services and software components (functions) communicate via a shared service broker. The middleware protocol used is called ARA and is inspired by Common API.

Figure 2: Adaptive AUTOSAR, including integration of Classic AUTOSAR software components
**Ethernet and TSN as an intermediary**

Most control units communicate with sensors and actuators via Ethernet. Time Sensitive Networking (TSN), an extension to the Audio Video Bridging (AVB) protocol, is used to implement safety-critical, reliable communication. The TSN standard was especially developed for safety- and real-time-critical systems such as ADAS (advanced driver assistance systems) and autonomous driving. In addition, Ethernet is used to connect infotainment systems to the internet and the vehicle manufacturer’s back-end system.

FlexRay is the loser of this change in technology. The fieldbus system is now applied by only a few OEMs and should be replaced soon. CAN and CAN FD (CAN with flexible data rate) will still be used to connect sensors and actuators or smaller IO (input/output) control units.

IO devices and the central computers communicate via a service-oriented interface specified in 2011 by the BMW Group: "scalable service-oriented middleware over IP", abbreviated as SOME/IP. It is based on Ethernet and the TCP/IP protocol family. The essential aspect is that SOME/IP automatically maps a defined application interface to packets. The advantage of SOME/IP is that it can even be integrated into small devices and enables quick starting of the overall system.

**Functional architecture**

Aside from the infrastructure issues presented above, largely specified by the AUTOSAR consortium and implemented in Elektrobit’s AUTOSAR product line EB tresos, the need for a functional architecture with defined interfaces between the function blocks is becoming increasingly clear. The advantage of common standardized interfaces is that certain blocks can be exchanged and the option to buy or offer them as a product.

**Figure 3** illustrates the architecture of the "open robinos" project from Elektrobit. The left-hand side shows the components for vehicle positioning and object fusion that combines objects detected by various sensors to form an overall picture. Trajectory planning, acceleration and steering angle are then determined based on the current driving situation.

**Figure 3: Elektrobit's "open robinos" architecture**

The objective of the project is to develop an open reference architecture that defines software components, interfaces and control mechanisms. The specification is freely available on the Elektrobit website and the project is open to OEMs, suppliers and partners. This approach is new to the market; however, its aim is to be integrated into different ADAS platforms. In this context, the platform is a part of different hardware products with different operating systems such as Adaptive AUTOSAR, QNX, or Unix-like OS that are available on the market.
Infrastructure as a challenge

The infrastructure where the vehicle moves is a major technical challenge for self-driving cars. Currently, vehicles are equipped with as many sensors as possible so that they autonomously find their way through endless different traffic situations. This approach is expensive and complex compared to trains and planes moving in a protected zone that is controlled externally. For example, altitude and route are directed by air traffic control services and trains are automatically stopped when they enter areas that are not open.

However, one cannot put a fence around all roads and bar cyclists. But road infrastructure modifications that tell the car, for example on entrance/exit ramps, that it is on a motorway and not on a parallel country road located some meters away would simplify the position detection problem. Another example is a parking structure that controls the vehicle remotely and guides it to an available parking space. This concept is simpler than vehicles that, looking for a free parking space, autonomously roam through the parking structure.

The prerequisites for these use cases are fast, nation-wide mobile data network (5G) rollout for data exchange with the back-end and infrastructure as well as the option to promptly adapt the road infrastructure to autonomous driving.

Which development process is emerging?

How can the requirements for both functional safety and - especially with regard to the increasing connectivity of vehicles - information security be met in these highly complex overall systems? To meet the high quality standards of automotive software development, the Automotive SPICE process model is well established throughout the industry. It forms the basis for safety and security.

The ISO 26262 standard defines how functional safety aspects can be implemented in system development at both the process level and the method level. For software architectures, functional safety is a key factor. Basic integrity mechanisms such as monitoring system integrity, partitioning, time and process monitoring or safe communication are available and are already being used in series projects.

Security mechanisms have been relevant in automotive development for quite a long time. Systems such as immobilizers, secure electronic keys or secure storage of the odometer are often already standard features. However, due to the increasing connectivity of vehicles, the industry is facing new challenges. According to the basic rule of information technology, "whatever is connected will be attacked", the system aspects of security and privacy are of growing significance in the automotive industry as well.

The first successful attacks on systems using remote access or the internet have already been made public and have evoked a wide response. In response, SAE International published a manual for the development of secure systems at the beginning of 2016 (SAE J3061, "Cybersecurity Guidebook for Cyber-Physical Systems"). It describes both processes and methods and follows ISO 26262 with regard to the life cycle. The document is not a standard. However, it summarizes essential efforts such as research programs or existing standards and publications. As such, it is a valuable contribution and can serve as an entry point to the introduction of security processes and methods.
Conclusion

The requirements for architectures for autonomous driving have become significantly more complex. However, by combining aspects such as standard architectures, functional safety, security, multi-core systems and availability, it is possible to design dependable systems and ideally evaluate and combine the individual system aspects depending on the use case.

All those involved in the automotive supply chain need to develop a core competency: systems engineering and therefore the interdisciplinary understanding of physics, electronics and software.

In the future, (software) developers must have a better understanding of systems in order to model the system behavior in the appropriate tools with linked code generators. Classic software development focuses on the development of tooling, code generators, and standard functions that are purchased as reusable products. Integrating the software will still require specialists who understand, analyze and fix errors at all levels from deeply embedded to service-oriented behavior.

In the coming years, new vehicle manufacturers and suppliers will appear on the automotive market. Especially IT companies have been using these technologies for years in other areas and follow the vision to run the car as a smartphone on wheels. The reason for this is that autonomous driving gives vehicle occupants considerably more time. This time can be used, for example, to use social networks, do online shopping, or work. Converting ‘driving time’ to ‘internet use time’ gives rise to entirely new business models.

OEMs' business cases, too, will be increasingly determined not only by selling, but also by running the vehicle. Among the ideas being discussed are novel rental and leasing concepts whose subject matter is no longer the car as a product but mobility as a service. In the future, the rental price for the autonomous transport capsule that takes you to work could differ depending on the time of day.

Rudolf Grave

studied electrical engineering at the Technical University of Hamburg-Harburg. He has been working for Elektrobit since 2005 and is responsible for AUTOSAR projects. His focus is on multi-core and functional safety.
About EB Automotive

Elektrobit (EB) is an award-winning and visionary global supplier of embedded software solutions and services for the automotive industry. A leader in automotive software with over 25 years serving the industry, EB's software powers over 70 million vehicles and offers flexible, innovative solutions for connected car infrastructure, human machine interface (HMI) technologies, navigation, driver assistance, electronic control units (ECUs), and software engineering services. EB is a wholly owned, independent subsidiary of Continental AG.