

Recipe for Safe Software

Development of ECU basic software according to ISO/DIS 26262



With the introduction of the new ISO 26262 standard, requirements for safety-related functions are becoming much more challenging than they used to be. At the same time, they are more precisely and clearly defined. Formally validated systems and the re-use of existing solutions do not need to contradict one another here. Generic safety modules in hardware and software can supplement proven components.

ISO 26262 is currently a draft standard in the voting phase but is expected to become a binding safety standard in the year 2011. It defines functional safety for electric/electronic (E/E) systems in an applicable manner. The future standard is based on the IEC 61508 standard that is used in a variety of industries, and it also addresses the fact that safety-related and non-safety-related functions are often interrelated in the automotive field, and therefore cannot always be clearly isolated from one another. The goal of ISO 26262 is to create a better understanding of safety-related functions and narrow their range of interpretation as much as possible.

One challenge in engineering development is to assess all potential hazards and risks in advance and take suitable measures to minimize these risks. To facilitate this, ISO prescribes that a "hazard and risk analysis" has to be performed at the beginning of development. All operating states and their associated failure types are analyzed here within the system under consideration, and various potentially hazardous situations are identified. Afterwards,

each hazard is assigned a safety requirement level (Automotive Safety Integrity Level, ASIL) from A to D, where A is the lowest and D the highest safety level.

As the ASIL increases, requirements increase for hardware metrics (FIT rates, test coverage, etc.) of the system as well as for the software development process (testing depth, requirements tracking, review documentation, etc.). The system supplier must fulfill these heightened requirements in addition to the high quality requirements which already exist.

Mastering rising complexity with standardization and re-use

In recent years, not only have the various in-vehicle functions grown in number and complexity; they are also more often being distributed throughout the vehicle. On the one hand, this trend was enabled by significant growth in computing power of the

processors used, and on the other by the larger bandwidth available in networking.

Meanwhile, the implemented functions have attained such complexity that conventional development methods can no longer meet the requirements of this architecture. This situation led OEMs to join together in the AUTOSAR development partnership in 2003; their common goal is to define a uniform software architecture for ECUs and decouple the hardware from the software.

Before AUTOSAR was defined, the communication stack and diagnostics were primarily the focus of automotive OEMs. With AUTOSAR, the basic software has undergone significant expansion. While it previously covered such areas as CAN, LIN, FlexRay and diagnostics, basic software now also includes the operating system, watchdog, memory stack and driver layer for the microcontroller.

AUTOSAR basic software attained a very high level of complexity when version 4.0 was released at the end of 2009. Over 80 software modules are defined via system description files (AUTOSAR System Configuration Description); they are also configured with powerful tools and their code is generated.

Requirements of AUTOSAR basic software used in safety-related ECUs

Aside from functional requirements, one key requirement remains: the basic software must not "disturb" the safety-related software. The meaning is two-fold: for one, the basic software must not violate the memory of the safety-related software, and for another, the basic software must not require more execution time than originally intended. Together, these conditions are also known as "freedom from interference".

The seemingly obvious approach is to develop the entire basic software based on standards for safety-related software grouped under ISO 26262. However, as already mentioned, the AUTOSAR basic software is very complex, has an enormous functional scope,

and its specifications are subject to short revision cycles. This would make development based on ISO 26262 very extensive.

Another approach is to reliably separate the basic software from the safety-related software without interference by implementing a protection layer (software partitioning). AUTOSAR already contains the functionality needed for this: memory protection and program flow monitoring. This functionality makes it possible to perform an ASIL decomposition (see following section) of the basic software according to QM and the protection layer of the desired ASIL according to ISO 26262. Therefore, it is sufficient to just develop the modules for memory protection and runtime behavior (program flow monitoring with watchdog) according to ISO 26262.

Use of ASIL decomposition according to ISO 26262

As already mentioned, ISO 26262 does not absolutely require re-development of all software mechanisms. To reach a certain ASIL, which is necessary to reach a defined safety goal, a combination of independent elements and suitable partitioning of safety requirements into redundant safety requirements can be used. For example, it is possible to attain ASIL C by combining two components with ASIL B and ASIL A while addressing specific requirements (Figure 1). ASIL decomposition can be performed on the system, hardware and software levels. In performing decomposition, care should be taken to ensure that the hardware architecture metrics remain unaffected and that there is no likelihood of violating the safety goal. In addition, ISO 26262 requires mandatory confirmation reviews that are based on the safety goals of the original ASIL (in our example: ASIL C).

To guarantee the necessary independence of the software components of different ASILs in an ECU, the standard defines rules for maintaining and verifying freedom from interference. These rules have to be kept and verified. That is because – in the special case of safety-related and non-safety-related elements – "coexistence" of the elements requires suitable measures and their verification.

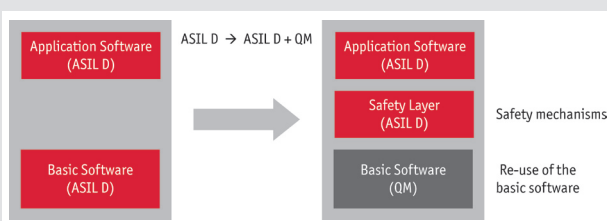


Figure 1: ASIL decomposition: A safety layer is added to the basic software to reach ASIL D.

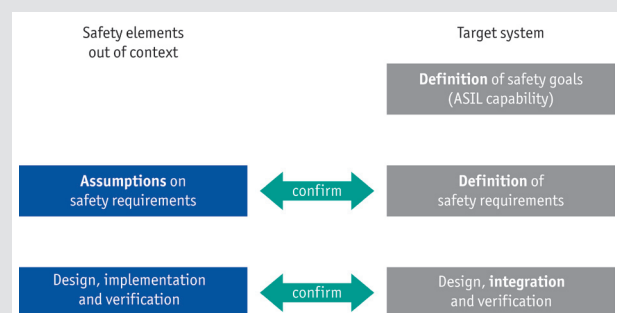


Figure 2: Prior to the development of a so-called "Safety Element out of Context" (SEoC) assumptions must be made and documented

Generic components (subsystems, hardware components, software components) that still do not have any concrete use case (“items”) during development - and therefore no safety goals are defined for them - can be developed according to ISO 26262 as so-called “Safety Elements out of Context” (SEoC).

In this case, since the applicable “safety requirements” to be fulfilled with the help of the element are unknown, “assumptions” must be made and documented accordingly (Figure 2).

When the SEoC is first used, a supplied Safety Manual must be checked to determine whether the assumptions that were made agree with the safety goals and safety requirements defined in the use case, its conformance is supported and no contradictions result. Correct use according to the safety manual must then be assured and verified.

Generic Implementation of the Validation Layer

TTTech Automotive has developed a generic monitoring layer in its “SafeExecution” product that fulfills the cited requirements for “coexistence” and freedom from interference and contributes to cost reduction by efficient reuse of existing components. With module integration for memory protection and program flow monitoring, it is possible to reliably detect potential errors in QM developed portions of the basic or application software, and suitable reactions to errors can be initiated (Figure 3).

Users must integrate the SafeExecution modules based on their safety manuals to achieve a system that satisfies the requirements of ISO/DIS 26262.

Besides its proven solution for end-to-end communication validation “SafeCOM,” SafeExecution is the second module from TTTech that is used to develop safety-related ECUs.

MICROSAR Safe as an Integrated Solution

To get validated basic software from a single source Vector and TTTech integrated the generic software modules SafeCOM and Safe-Execution into MICROSAR – the practice-proven AUTOSAR solution from Vector (Figure 4).

Re-use of certifiably developed central software components reduces costs for integration of the application. Instead of an application-specific solution, TTTech and Vector are together offering a generic standard solution that minimizes development costs for ISO 26262 conformant safety-related ECUs.

Translation of a German publication in Elektronik automotive, 11/2010

All Figures: Vector Informatik and TTTech Automotive

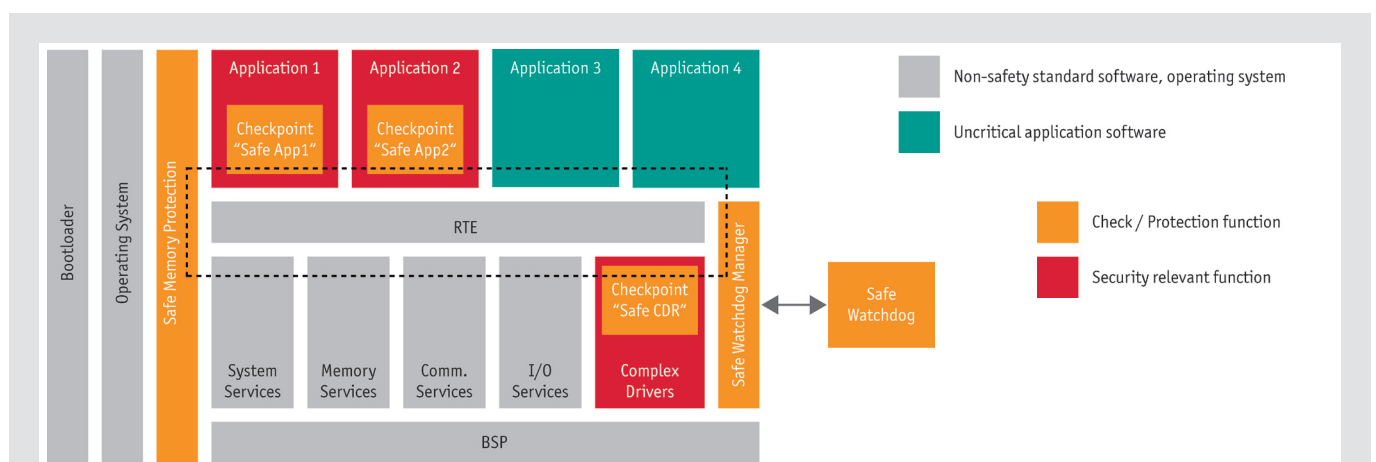


Figure 3: The SafeExecution modules must be integrated in an existing system according to the safety manual to fulfill the requirements of ISO 26262

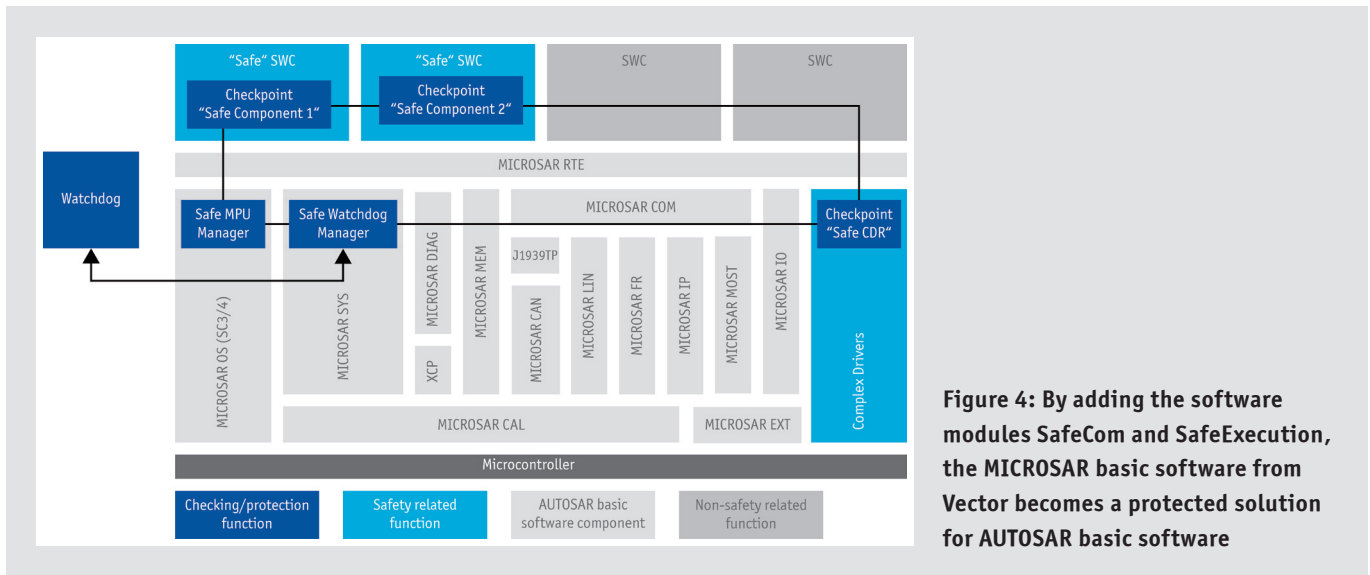


Figure 4: By adding the software modules SafeCom and SafeExecution, the MICROSAR basic software from Vector becomes a protected solution for AUTOSAR basic software



Joachim Kalmbach (Dipl.-Ing. (FH)) studied Computer Science and Automation at the University of Applied Sciences at Reutlingen. In 2006, he joined Vector Informatik GmbH in Stuttgart where he is currently a Product Manager in the Embedded Software area. His work focuses on AUTOSAR and Functional Safety.



Dr. Thomas Wenzel Since 2006, he has been working at the Institute for Vehicle Technology and Mobility, where he has been managing the competency field of Functional Safety since March 2010. Before assuming this position, he earned his doctorate degree in Control Engineering at Coventry University.



Martin Fassel is a Graduate Engineer in Engineering Physics (Technical University of Vienna); since 1997 he has been working in hardware and software development in the telecommunications, aerospace and automotive fields. Since 2008, he has been working at TTTech Automotive GmbH as Product Manager for Development Tools and Standard Software.

>> Your Contact:

Germany and all countries, not named below

Vector Informatik GmbH, Stuttgart, Germany, www.vector.com

France, Belgium, Luxembourg

Vector France, Paris, France, www.vector-france.com

Sweden, Denmark, Norway, Finland, Iceland

VecScan AB, Göteborg, Sweden, www.vector-scandinavia.com

Great Britain

Vector GB Ltd., Birmingham, United Kingdom, www.vector-gb.co.uk

USA, Canada, Mexico

Vector CANtech, Inc., Detroit, USA, www.vector-cantech.com

Japan

Vector Japan Co., Ltd., Tokyo, Japan, www.vector-japan.co.jp

Korea

Vector Korea IT Inc., Seoul, Republic of Korea, www.vector.kr

India

Vector Informatik India Prv. Ltd., Pune, India, www.vector.in

E-Mail Contact

info@vector.com