Safe Automotive Software

Karl Heckemann[1], Manuel Gesell[2], Thomas Pfister[3], Karsten Berns[3], Klaus Schneider[2], and Mario Trapp[1]

[1] Fraunhofer Institute for Experimental Software Engineering, Fraunhofer-Platz 1, 67663 Kaiserslautern, Germany
{karl.heckemann,mario.trapp}@iese.fraunhofer.de
http://www.iese.fraunhofer.de

[2] Embedded Systems Group, Department of Computer Science, University of Kaiserslautern
{gesell,klaus.schneider}@cs.uni-kl.de
http://es.cs.uni-kl.de

[3] Robotics Research Lab, Department of Computer Science, University of Kaiserslautern
{pfister,berns}@cs.uni-kl.de
http://agrosy.cs.uni-kl.de

Abstract. For automotive manufacturers and tier-1 suppliers, the upcoming safety standard ISO 26262 results in new requirements for the development of embedded electronics and software. In particular, the variety of driver assistance systems that autonomously influence the driving dynamics of a vehicle may have a high risk potential and require development in accordance with the normative guidelines. But especially for those systems whose function is typically not based solely on hardware but on complex software algorithms, safety certification can be very complex or even impossible. In this paper the problems of development of vehicle systems according to ISO 26262 are described. Finally an approach for a safety-oriented reference architecture is presented that introduces adaptive software safety cages. This architecture enables application of formal verification methods. Supported by multisensor data fusion this allows to reduce safety requirements for vehicle control systems.

1 Introduction

According to car manufacturer's development road maps, vehicles will be able to drive autonomously by the year 2030 [9]. This goal represents significant technical challenges to electronic systems in vehicles. The trend towards autonomy is already visible in the development of today's driver assistance systems. Until a few years ago, cruise control systems were only capable of maintaining a fixed vehicle speed. Current adaptive cruise control systems however allow to adjust to the speed of the vehicle in front. The emergency braking system further enhances this function by allowing the system to initiate a full brake in case of sudden braking of the vehicle ahead or appearance of an obstacle. Multiple other semi-autonomous systems have been introduced in modern cars, such as "automatic

A. König et al. (Eds.): KES 2011, Part IV, LNAI 6884, pp. 167–176, 2011.

parking". These driver assistance systems take control of the vehicle dynamics without the intervention of the driver. The further development towards full autonomy of the vehicle creates the need for systems that allow to process an extensive amount of input data representing the vehicle condition and the vehicle environment.

From the safety point of view two problematic trends emerge:

1. Today, the driver is considered to be part of the safety concept. Thus, a malfunction of a system is classified as less serious if the driver can control the vehicle in spite of the malfunction by overriding the faulty system. However, the goal of autonomous and semi-autonomous driving functions is making driving manoeuvres *without* an intervention of the driver. Hence the safety relevance of the respective functions increases with the degree of autonomy.
2. The system complexity is growing steadily as functionality of vehicle systems increases. The more complex a system is, the more difficult it is to verify its freedom of faults and thus to guarantee its safety. This is especially true when considering intelligent systems like autonomous vehicle systems.

The consequence of these trends is that more and more safety-critical functions in the vehicle rely on systems whose correct behaviour can not be guaranteed by traditional approaches.

In contrast to aviation, where a uniform standard for functional safety has been in use for years, there has been no specific standard for the automotive industry until now. The application of other industry safety standards is problematic due to specific properties of the automotive industry, like mass-production and cost pressure, which excludes safety measures that are common to other industries. A classic approach, which is used for example in avionics, is redundancy of safety-critical systems. Due to associated production costs and space limitations this is unreasonable for road vehicles.

After the publication of the final version of the ISO 26262 standard [12] in mid-2011, it will be the agreed state-of-the-art of technology for road vehicles up to 3.5 tonnes. ISO 26262 is an adaptation of the generic functional safety standard IEC 61508 tailored to the automotive industry . The aim is to reduce systematic and random faults in electric and electronic vehicle components and thus to increase car and road safety.

ISO 26262 does not specifically address intelligent systems. Still there are certain requirements that might collide with the development of artificial intelligence, especially in the domains of specification, design and testing.

The standard requires that already in the specification phase measures are taken to ensure testability of the software. Regarding the architectural design, modularity, encapsulation and minimal complexity are required. Also on system level, avoidance of unnecessary complexity is demanded. Intelligent systems contributing to (semi-) autonomous driving necessarily depend on a variety of sensor data. In order to evaluate these and to determine the current driving situation and vehicle environment, complex processing is necessary. This contradicts the requirement of low complexity on software and system level.

A system level requirement of ISO 26262 is that the "response of the system or elements to stimuli, including failures, and relevant combinations of stimuli, shall be specified for each technical safety requirement". This is a problem particularly for non-deterministic systems. The behaviour of an optical object recognition system, for example, can not be predicted as the set of possible input data in form of camera images can never be considered completely. In practice, input data of this kind can not even be reproduced exactly. Thus sufficient test coverage of such a system can never be claimed.

In [4] means to attain dependability are defined, that describe ways to prevent system failures in different states of the product lifecycle: fault prevention, fault tolerance and fault removal. It is not possible to address all malfunctions of a system by the means of fault prevention and fault removal, because of the complexity of the system and the non-determinism of the environment. Hence, fault tolerance is obviously the only applicable method. One way of implementing this is heterogeneous redundancy, that ISO 26262 supports by the means of ASIL (automotive safety integrity level) decomposition.

In this paper, we present a new approach of a safety-related reference architecture for road vehicles following the ISO 26262 standard. This architecture allows to overcome the normative problems associated with today's and future road vehicle systems by providing an independent safety mechanism, called safety cages. Our safety cage architecture allows a formal verification of the system behaviour.

The rest of this paper is structured as follows: Section 2 describes how safety is currently handled by the state-of-the-art framework AUTOSAR and also the Simplex architecture, a motivation for our approach. Section 3 forms the core of our paper: it introduces the adaptive safety cage architecture as a framework for ISO 26262 conform development of complex vehicle systems. Finally, we summarize our work and give an outlook to future tasks in Section 4.

2 State of the Art

In the automotive context, standard software systems concerning functional safety focus rather on the detection of hardware faults than the functionally correct implementation of software [3]. For instance AUTOSAR, a specification that defines an automotive open system architecture, addresses safety for the first time in version 4.0. In this version it defines features for hardware fault detection, like RAM testing, flash memory testing, CPU core testing, support for ECC memory and watchdog. Additionally, the specification defines several measures to ensure that software is executed as intended, like program flow monitoring, timing and synchronization features, communication protection and memory partitioning. AUTOSAR aims at providing a software platform for hardware independent development of applications, called software components (SW-C). It does not define any software components, but only the execution environment by abstracting hardware dependent features and communication methods. This explains why the defined safety features focus on the correct execution of software

components only, and why monitoring of functional behaviour of the application is neglected.

In [14], Sha argues that diverse redundancy, particularly N-version programming, often leads to reduced software reliability. The reason is that development efforts have to be divided due to a limited budget, resulting in reduced effort put into the development of each version. As an alternative a heterogeneous redundancy architecture called Simplex architecture is presented. It consists of a plant that is to be controlled, a control system divided into two subsystems and a decision logic. One of the subsystems is a high-performance-control subsystem (HPC), that is controlling the plant in regular operation. In contrast to the HPC, the high-assurance-control subsystem (HAC) is a simple and conservative control system. The decision logic monitors the plant's state and switches from the HPC to the HAC if the plant's stability is impaired.

3 Adaptive Safety Cage Architecture

Traditionally, the primary objective of safety development is not to prevent malfunctions of the system itself, but to ensure that safety mechanisms are provided that identify faults and provide countermeasures to prevent damage to persons and environment. For example, a suitable safety measure for a robotic arm that is capable of harmful movements is a physical safety cage, that limits the range of motion of the robot to its work area. Such a safety cage can be implemented as a light curtain, which invokes an emergency stop, once it is broken. The focus of the safety-related development can then be limited to the safety mechanism itself.

In the automotive context, today's highly software-controlled electronic systems are typically not divided into a function and a safety mechanism. Instead, the usual approach of acquiring confidence of the correctness of an automotive function is to develop the function itself according to a safety standard. As seen before, it will be difficult to achieve this for intelligent systems with ISO 26262 as the mandatory standard.

The SafeCAr project (Adaptive Safety Cage Architectures) aims to transfer the traditional approach of functionally independent safety mechanisms to software-based automotive systems. Referring to the motivation example, we use the term "software safety cage" to describe the explicit safety mechanism. A software safety cage is a piece of software that monitors the behaviour (outputs) of the original function and takes appropriate actions if a malfunction is detected. Using these software safety cages we develop a reference architecture that can be adapted by vehicle manufacturers and tier-1 suppliers to simplify the safety-related development process.

Like the above mentioned physical safety cages, our software safety cage separates two areas of operation: a valid area, that is considered safe and needed to fulfil the function and an invalid area, that could lead to a dangerous situation. As the valid area of operation for a robotic arm is defined in physical space, in the context of vehicle dynamics, an n-dimensional state space must be considered. The dimensions of this space are determined by the degrees of freedom of

the vehicle (e. g. speed and steering angle) and its environment (e. g. distance to objects and road conditions). If the state space is complete, meaning all degrees of freedom are included, every possible state of the vehicle can be represented as a point in the state space. In combination with vehicle dependent fixed parameters, in this state space areas of vehicle dynamic stability (valid area) and instability (invalid area) can be defined. Ideally, a vehicle safety cage would be able to exactly measure all these dimensions and disallow vehicle systems to perform state transitions to an invalid area of operation. However, in practice we have to face several limitations:

- Since sensors are not perfect, there is always a deviation between the true and the measured value. The safety cage approach addresses this using multisensor data fusion (see 3.4).
- With the large number of inputs the safety cage itself becomes very complex. This problem is addressed by splitting up the overall vehicle safety cage to multiple smaller safety cages, that address only certain aspects of the system (see 3.1). The focus on a subset of degrees of freedom also simplifies the otherwise tedious definition of valid areas of operation.
- There are settings in which a system malfunction leads to a dangerous situation, although driving dynamic stability is not affected. For example, an emergency braking system might brake due to a false detection of an obstacle. As correctness of object recognition can not be proved, there is no way to prevent this kind of hazard. Still, we try to mitigate the outcome of such system malfunctions by the means of context aware safety cages (see 3.2).

Once a failure is detected, an appropriate action has to be invoked by the safety cage. As the objective is to create a fault tolerant system, the faulty subsystem should not be simply deactivated. Instead, the system state should be transformed to a state without detected errors and faults [4]. This transformation is called recovery. One way to achieve this is graceful degradation, which means that a system is able to reduce its functionality or its performance (or both) in order to keep running in spite of the occurrence of a failure. In the safety cage context, a graceful degradation could be triggered by the safety cage as a reaction to detected failures.

The normative concept behind the software safety cages is ASIL decomposition, as described in [12]. This means the ASIL assigned to system elements can be lowered, if redundancy is used to achieve one safety goal. Considering one defined safety goal, the safety cage acts as a functionally redundant system to the actual control function. In the ideal case the ASIL of the safety goal would be assigned to the safety cage. Then the automotive system only has to be developed according to the quality management standard. To fulfil safety requirements the correct function of the safety mechanism must be guaranteed. To prove the correctness of the safety cages, formal verification methods are to be used (see 3.5).

The rest of this section is structured as follows: First, we will classify different kinds of software safety cages (Section 3.1) and describe the need of context awareness (Section 3.2). Next, an outline of the target architecture is given,

regarding hardware and software issues (Section 3.3). Then sensor fusion and failure detection are described (Section 3.4). Finally we discuss the problems of vehicle system and safety cage verification (Section 3.5).

3.1 Classification

To handle the complexity of determining the state of the vehicle using a multitude of sensors, it seems reasonable to implement a vehicle safety cage as a combination of multiple domain specific safety cages that are focussed on certain aspects of the system. Therefore domains as well as application points for these safety cages must be defined. The basic control loop of vehicle dynamics control systems consists of three elements: *sensors*, that detect certain aspects of the current driving situation, *electronic control systems*, that process the sensor data and generate actuator commands accordingly and *actuators*, that influence the driving dynamics and as such the driving situation. If we consider sensor outputs as reliable (see 3.4), we can distinguish two application points for different safety cage classes: the control systems and the actuators.

Functional safety cages cover a single control system functions or a group of control system functions that share certain characteristics. A possible domain for this class of safety cages could be low-speed functions, like automatic parking, traffic jam assistant, electric parking brake or hill holder. This allows the safety cage to be tailored to the characteristics of this specific domain.

Actuator safety cages are focussed on the immediately safety-related actuators in the vehicle that have a direct influence on the longitudinal and lateral dynamics: brakes, steering and engine control. Using a safety cage to guarantee that actuators are controlled within defined limits corresponding to the vehicle state, safety requirements of a whole group of safety related vehicle control systems can be covered.

3.2 Context Awareness in Safety Cages

Not every hazardous event potentially caused by a vehicle system is related to driving dynamic stability. Emergency braking, as an example, may not impair stability, still it can be a dangerous manoeuvre in certain situations. Therefore during hazard and risk analysis all combinations of hazards and driving situations are evaluated. These combinations are then assigned an ASIL based on their estimated levels of exposure, severity and controllability. Looking at emergency braking, obviously the severity is higher in a high speed driving situation, also resulting in a higher ASIL. By means of context awareness, a safety cage can check the plausibility of function outputs and restrict or block functions depending on the current driving situation, determined by information such as vehicle speed, driving dynamics or GPS location (urban traffic/highway). Applied to the example of the emergency braking system this means that emergency braking due to a detected pedestrian is allowed in urban areas, while it is restricted to a limited braking power at high speeds on a highway.

3.3 Architecture

Current vehicle networks consist of a multitude of interconnected electronic control units (ECU), each running software to provide one or several vehicle functions. The different components of the safety cage architecture can be implemented as an additional ECU. Alternatively they can be integrated in the existing ECUs, requiring independence to guarantee freedom of interference. While the second case uses ECU-internal communication, a separate safety cage ECU communicates using the vehicle's communication bus. To ensure that all commands to actuators pass the safety cage, it has to be realized as a gateway between separate communication busses of functions and actuators.

As depicted in Fig. 1, the safety cage architecture consists of a combination of the different safety cage classes. The vehicle functions (F_1 to F_n) work within functional safety cages. Their outputs are processed by a filter and a gateway. The filter blocks all commands that can be statically defined as invalid, for example by a white list that defines which function is allowed to access which actuator. The gateway separates the communication busses of functions and actuators. It is able to dynamically decide what function is allowed to access actuators depending on the current driving situation and thus represents a context aware safety cage. For example, a braking request of a low-speed function can be suppressed if the vehicle speed exceeds a certain level. As the final elements of the safety cage chain, the actuator cages are arranged logically in front of the actuator control systems and check the plausibility of the requested actuator values.

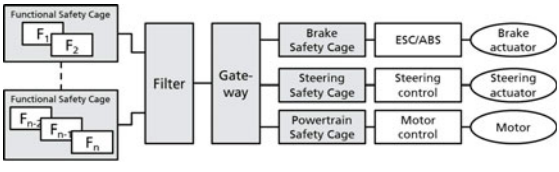

Fig. 1. Components of the safety cage architecture

3.4 Sensor Fusion and Failure Detection

When dealing with safety critical applications one has to face the characteristics of the employed sensors. There are two important aspects which have to be considered. First the reliability of the sensors and second, the imperfection of the measurements. Like every other technical device a sensor can break or fail. All sensor outputs are in some way disturbed by sensor noise that leads to unpredictably corrupted measurement of the true value.

For safety critical automotive applications these two aspects are of high importance. The fact that applications are based on unreliable data causes them to be unreliable as well. To improve the degree of reliance in critical systems the usual approach is to employ redundancy. In order to address this approach

voting algorithms are commonly applied. Even though this eases the problem it causes a dramatic increase in cost at the same time. As an alternative to the redundancy and voting approach there are some techniques that facilitate fault detection and isolation (FDI). The goal is to detect an occurred failure and to isolate the faulty component. These techniques employ many different methods and originate from a broad variety of research areas [15,16,18].

To increase the precision of a single sensor filtering methods can be utilized. If redundant sensors are available or the relation of sensors can be described through a mathematical model, multisensor data fusion is a way to do so. The task of fusion is to combine the outputs of several sensors in one representation. Additionally to the better accuracy, a higher reliability and temporal availability can be achieved. Most fusion algorithms are based on probabilistic methods [8]. They are favored because of their ability to model and handle the uncertainty of measurements.

A very popular family of probabilistic fusion algorithms are Bayesian filters, e. g. Kalman filter or particle filter. Those filters are thoroughly investigated and used for fusion problems. In their basic formulation they suffer from missing ability to handle faulty sensors. Some approaches have been proposed to overcome this drawback [6,10]. In the context of safety cages, multisensor data fusion will be used to improve the knowledge about the system. In contrast to a vehicle system, the safety cage could be provided with additional information about the employed sensors. Besides the more accurate fusion result, an appropriate failure detection indicator for the sensors should be delivered. For this purpose different approaches for failure detection in fusion algorithms have to be tested and new techniques should be developed. With these additional information the safety cage is enabled to avoid erroneous function of the application originating from faulty or inaccurate measurements.

3.5 Safety Cage Verification

Today's vehicle systems are hybrid systems [11]: they consist of continuous and also of discrete parts. A high ASIL classification of these systems requires the usage of verification techniques for hybrid systems [1]. An approach for verification by abstraction and discretization of the continuous part is done in [2,5]. However, it comes as no surprise that this kind of verification has to deal with a large state space for industrial-sized systems. This also holds for vehicle systems, which are usually very complex.

As mentioned, we need to reduce the ASIL classification for these vehicle systems. For this reason, our approach introduces less complex functional safety cages that logically enclose one or more vehicle systems. This leads to a high ASIL classification for the safety cages, but a low classification of the enclosed systems. The safety cage supervises the included systems and ensures the feasibility of the outputs and trigger reconfigurations or shutdowns of the included systems in case of unreliable or faulty sensor data. The multi sensor data fusion will determine the quality of sensor data. Fortunately, the safety cage is much simpler than the original control system and is realizable as a discrete

system. Hence, traditional verification techniques are applicable to ensure the correctness. We will use model checking or invariant checking techniques [7] to ensure that the introduced safety cages are an adequate abstraction and discretization of the enclosed vehicle function and ensure the correct functioning of the composition. We will use Matlab/Simulink models to describe our safety cages, because already existing vehicle systems are described in that way and the used simulator and demonstrator tool for this project is based on Matlab/Simulink. Therefore, it is necessary to add the ability to apply the above mentioned formal verification techniques to those models. We plan to use the SMV System[13] for model checking purposes and the theorem prover Isabelle/HOL[17]. Both are state of the art tools in their application area and we already have first-hand operating experience with these tools in other projects. Thus, we have to develop an appropriate transformation/abstraction of Matlab/Simulink models to these tools.

Another task is to verify the actor safety cages to ensure that the correct behaviour of all safety cages implies a correct behaviour for the whole system. Invariant verification techniques will be used to avoid a state space explosion. Necessary modular abstraction techniques to enhance the verification will be derived as well in this stage. To this end, we consider the following steps as a reasonable proceeding: During the development of the first models we will fix an exact definition of a suitable and supported specification language to start all verification efforts. A translation technique for Matlab/Simulink models to SMV models and Isabelle/HOL theories must be developed immediately. Later, we will determine specialized abstraction and translation techniques for our purposes.

4 Summary and Future Work

In this paper, we presented a new approach of a safety-related reference architecture for road vehicles following the ISO 26262 standard. The safety cage architecture allows a formal verification of the system behaviour, and therefore increases reliability. The use of multisensor data fusion further improves dependability. This combination enables the development of safe automotive systems.

Future work in this project includes developing a simple model based on Matlab/Simulink to prove the concept's principles. Limiting the complexity of the initial system allows quick development cycles and an early evaluation of different design decisions. This model will be extended to a case study in which a basic safety cage is modelled for a driver assistance system. The intention of the case study is to determine in which way a practical application of the concept is possible. For simulating the resulting system we will use a commercial Matlab/Simulink based vehicle simulator (CarMaker) that allows to demonstrate the advantages of our approach. The long term goal is an implementation under real vehicle conditions.

References

1. Alur, R., Henzinger, T., Ho, P.H.: Automatic symbolic verification of embedded systems. IEEE Transactions on Software Engineering (T-SE) 22(3), 181–201 (1996)
2. Alur, R., Henzinger, T., Lafferriere, G., Pappas, G.: Discrete abstractions of hybrid systems. Proceedings of the IEEE 88(7), 971–984 (2000)
3. AUTOSAR: Technical Safety Concept Status Report (2009)
4. Avizienis, A., Laprie, J.C., Randell, B.: Fundamental concepts of dependability. In: Proceedings of ISW 2000, 34th Information Survivability Workshop, pp. 7–12. IEEE, Los Alamitos (2000)
5. Bauer, K., Gentilini, R., Schneider, K.: Property driven three-valued model checking on hybrid automata. In: Ono, H., Kanazawa, M., de Queiroz, R. (eds.) WoLLIC 2009. LNCS, vol. 5514, pp. 218–229. Springer, Heidelberg (2009)
6. Chetouani, Y.: Fault detection by using the innovation signal: application to an exothermic reaction. Chemical Engineering and Processing 43(12), 1579–1585 (2004)
7. Clarke, E., Grumberg, O., Long, D.: Model checking and abstraction. ACM Transactions on Programming Languages and Systems (TOPLAS) 16(5), 1512–1542 (1994)
8. Durrant-Whyte, H., Henderson, T.C.: Multisensor data fusion. In: Siciliano, B., Khatib, O. (eds.) Springer Handbook of Robotics, pp. 585–610. Springer, Heidelberg (2008)
9. GM: GM unveils EN-V concept: A vision for future urban mobility. Website (2010), `http://www.gmexpo2010.com/en-v/en/introduction/press` (visited on April 29, 2011)
10. Hajiyev, C.: Testing the covariance matrix of the innovation sequence with sensor/actuator fault detection applications. International Journal of Adaptive Control and Signal Processing 24(9), 717–730 (2010)
11. Henzinger, T.: Verification of Digital and Hybrid Systems. In: Verification of Digital and Hybrid Systems. NATO Advanced Study Institute Series F: Computer and Systems Sciences, vol. 170, pp. 265–292. Springer, Heidelberg (2000)
12. ISO/DIS 26262: Road Vehicles, Functional Safety Part 1 to 10 (2008)
13. McMillan, K.: The SMV system, symbolic model checking - an approach. Tech. Rep. CMU-CS-92-131, Carnegie Mellon University (1992)
14. Sha, L.: Using simplicity to control complexity. IEEE Software 18, 20–28 (2001)
15. Venkatasubramanian, V., Rengaswamy, R., Kavuri, S.N.: A review of process fault detection and diagnosis: Part ii: Qualitative models and search strategies. Computers & Chemical Engineering 27(3), 313–326 (2003)
16. Venkatasubramanian, V., Rengaswamy, R., Yin, K., Kavuri, S.N.: A review of process fault detection and diagnosis: Part i: Quantitative model-based methods. Computers & Chemical Engineering 27(3), 293–311 (2003)
17. Wenzel, M., Paulson, L.C., Nipkow, T.: The isabelle framework. In: Mohamed, O.A., Muoz, C., Tahar, S. (eds.) TPHOLs 2008. LNCS, vol. 5170, pp. 33–38. Springer, Heidelberg (2008)
18. Zhang, Y., Jiang, J.: Bibliographical review on reconfigurable fault-tolerant control systems. Annual Reviews in Control 32(2), 229–252 (2008)