Reliability-based safeguarding of automated traffic systems by means of "fail-operational" capable on-board architectures

Johannes Heinrich
Institut für Qualitäts- und Zuverlässigkeitsmanagement GmbH
Wuppertal, Germany
heinrich@iqz-wuppertal.de

Dr.-Ing. Fabian Plinke
Institut für Qualitäts- und Zuverlässigkeitsmanagement GmbH
Hamburg, Germany
plinke@iqz-wuppertal.de

Prof. Dr.-Ing. Andreas Braasch
Institut für Qualitäts- und Zuverlässigkeitsmanagement GmbH
Wuppertal, Germany
braasch@iqz-wuppertal.de

***Abstract*— Fully automated and autonomous vehicles place new demands on reliability, availability and safety. Eliminating the driver as a fallback path in the event of technical breakdowns or failures, forces a self-driving vehicle to operate safely even in the event of a fault, in order to reach a risk-minimum state. Similarly, minor disturbances, such as software crashes or failures, must be compensated for safety terms in real time. These requirements are realized through redundancy-based, fail-operational on-board architectures. In the event of an error, a fault-tolerant subsystem can be switched on or off in order to obtain a (possibly degraded) operability. In aviation, such system designs have been standardized and steadily developed for the use of complex, software-based flight control systems. The article gives an overview of the technical requirements of the aviation and automotive industries, as well as the presentation of aviation methods and principles and their transfer to the automotive development. This includes methods for fault detection, fault tolerance, strategies for continued operation, live repair, degradation and the safe shutdown of the system into a state of minimal risk. This procedure has not been implemented in the current automotive development because the responsibility for driving the vehicle lies always with the driver. A quantitative assessment of reliability, availability and safety – considering the above system properties – can be performed using multi-stage simulation models, which are also presented in this paper. The goal is the statistical validation of an economic system design while complying with the safety requirements as established by common standards and rules in the automotive sector (for example ISO 26262, SOTIF via ISO / PAS 21448, etc.).**

Keywords—availability increase due to fail-operational architectures, reliability of complex hard- and software architectures, multi-level simulation models

I. Introduction

The development of automated and autonomous systems is currently being heavily promoted across many industries, producing new results at regular intervals. A large share of this lies within the automotive industry, but also industries such as the railway industry or the process engineering research in these areas and try to use new technologies for themselves.

One aspect which, in addition to the pure technical implementation of the systems, represents a major challenge is the safe design of the systems and the safety proof to be provided. As a result of the elimination of humans as a control level and the assumption of control by software, the demands placed on the systems regarding safety and availability increase. In particular, the handling of failures and the associated fault-tolerant design of the system have a major role. The person who, in the case of an error, e.g. in the car, has to intervene, is no longer tangible, so that this fallback path must be replaced.

In this paper, on the one hand, chapter 2 presents concepts on how to increase the reliability of a system by means of redundancy, on the other hand, with the help of the Monitor-Control principle a possibility is shown with which the safe operation of the system can be ensured. In addition, Chapter 2 explains the possibility of degradation strategies that can be used to deactivate unnecessary functions in the event of a failure, so that sufficient capacity is available for safety-relevant functions. Chapter 3 presents first the problems that arise as a result of the increasing share of software and the dynamic distribution of functions in safety and reliability proofing in automated and autonomous systems. On this basis, based on the Monte Carlo Simulation, a model is presented with which software-loaded dynamic system architectures can be analysed with regard to adapted evaluation criteria. The possibility of automatically classifying failures in a vehicle and, accordingly, autonomously treating them does not currently exist in the field of automotive. A model from the field of aerospace is presented in Chapter 4 which addresses this topic and shows first approaches.

II. Requirements for safety-relevant system architectures

The safety and reliability concept of future automated and autonomous systems must be renewed due to the loss of a human person. Today's systems are often designed by using the fail-safe-principle, where a person can immediately take over the control of the system in case of a failure that lead to a function shutdown. Besides that, the person acts as a controller in terms of automated functions and must intervene in the event of a wrongdoing. For example, a driver has to monitor an automated car up to SAE Level 2 "Partial Automation" [1] and is responsible for the action of the system. Therefore, there is no need for fallback modes in terms of the system design to maintain or supervise the function. Automated driving functions of Level 3 "Conditional Automation" do not need a driver in the loop, but the driver has to be available as a fallback, e.g. in case of failure, after a takeover time. The system therefore has to remain capable of manoeuvring in the event of a fault within a defined time interval, it must be temporary fail-operational.

In case of Level 4 “High Automation” and Level 5 “Full Automation” the driver does not need to be in the loop and be available as a fallback, so there are special fault tolerance requirements for the system architecture due to the fail-operational-principle.

Besides the fault tolerance the system has to supervise itself in terms of safety-relevant actions, no accidents, especially with human damage, due to a malfunction can be tolerated.

Because of the described challenges in automated and autonomous systems suitable strategies must be used in order to guarantee the maximum of the safe state probability (availability of safety). A possible approach is to use the aerospace principles for Monitor-Control and redundancies.

A self-resolving system must be monitored in real-time. Pilots are monitoring the system in an aircraft. However, autonomous systems must be using redundancy to implement a Monitor-Control-Principle (Fig. 1).

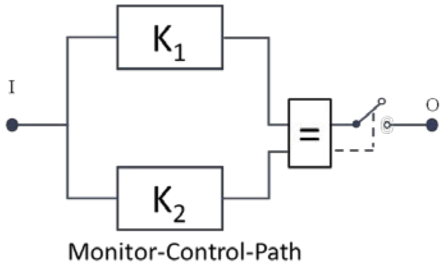

Fig. 1 General Monitor-Control architecture

A difference in the reliability and safety redundancy need to be considered. Simple parallel system increases the reliability but in order to use a redundancy with a Monitor-Control-Principle a m-out-of-n, n-out-of-n-systems or serial system with a voter is required, the parallel system can only detect a failure, but cannot determine which component causes the failure and which component is doing right.

The implementation of redundancy is more complicated and leads to an increase of redundant and partly dissimilar systems. However, on the one hand, economic aspects and, on the other hand, also structural aspects have to be considered in order to find a suitable solution.

In case of failures, where the system needs to access a safe failure state, the availability of base function needs to be ensured (probably without requirements to the redundancy, etc.). Therefore degradation strategies must be defined in order to use the available resources efficiently (Fig. 2).

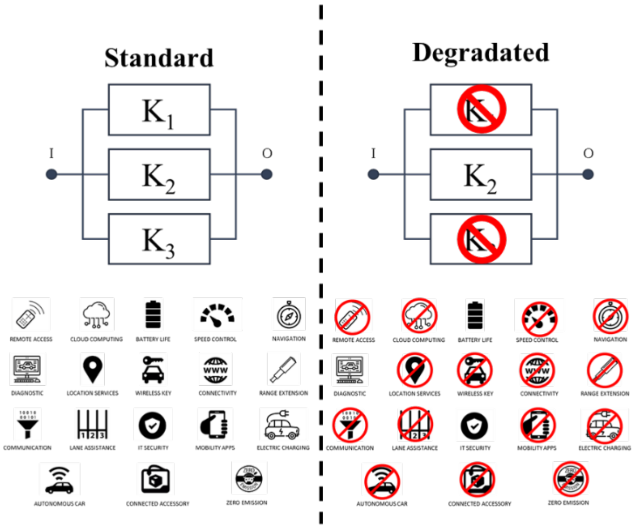

Fig. 2 Exemplary degradation concept

On the basis of these strategies, in the event of an failure, functions that are not necessary in the present situation can be switched off and the freed-up resources used for safety-relevant applications. For example, it would be possible to hijack a control unit which is used for functions in the area of infotainment in the event of failure of a control unit having a safety-relevant driving function and to be able to continue operating the function on it. Maintaining the safe onward journey would have priority. For the realization of such a concept it is necessary that the functions are provided with different priorities, which can change depending on altered use cases.

From the view of reliability engineering different paradigms are considered: reliability, availability, safety and the parameters are assessed for deviations. So far, the classical quality management often only determines reliability parameters and assigns them to safety critical scenarios.

III. Safety and Reliability Proof

Another challenge of automated and autonomous systems are the safety and reliability proof.

The classic reliability system structures for the conventional mechanical and electrical dominated systems cannot describe the whole picture of mainly software-based infrastructures [2]. The main reason for that is the dynamic of the software application and their failures and repair. After a software failure the components can be switched off, restarted or the whole application can be moved to another processor, resulting in a new repair and failure behavior that must be methodically defined. A new start of an application, a flash of an image or external repair processes via WLAN or mobile networks are examples that must be considered for the analysis. As a result of these boundary conditions, the parameter of availability of dynamic repairable systems is becoming increasingly important.

In quantified methods for reliability analysis the possibility for considering repair rates and repair behavior are already possible. However, describing the repair behavior for components realistically cannot be considered by classical methods, yet. The implementation of strategies, e.g. Markov-Processes or Petri-networks is mathematically challenging or even in some cases impossible.

Through the increasing realization of functions by means of software and the existence of redundant structures a temporary or permanent shift of a whole function implemented in software to a different hardware is possible and necessary [3]. The result is a change of the system structure during operation, which is adapting its tasks and modes according to fixed rules. A reliability optimization problem is created, which must consider the technical and economic efficiency of the system's design. Furthermore, safety requirements can depend on different use cases. Different requirements for the computing power of an automated or autonomous function exist depending on the size of existing influencing factors and the resulting complexity of the function. The system must be technically flexible to adapt to different requirement and environments. A solution is that dynamic system structures are introduced that can adapt their structure automatically according to the use case. For the reliability engineering multi-layered calculation on different levels must be done and summarized. This point poses another challenge to the reliability and safety analyses, since such dynamics combined with the repair behavior of the software can hardly be represented and calculated analytically.

This section presents an approach to handle the above-mentioned challenges in terms of safety and reliability proof. A possible solution to represent the boundary conditions is to use the Monte Carlo Simulation (MCS) [4] [5] [6] [7] whereby the systems can be modelled very flexible.

The simulation is based on state diagrams in which the system is mapped with regard to active and failed hardware and software. The system, which is calculated by way of example in this paper, is shown in Fig. 3.

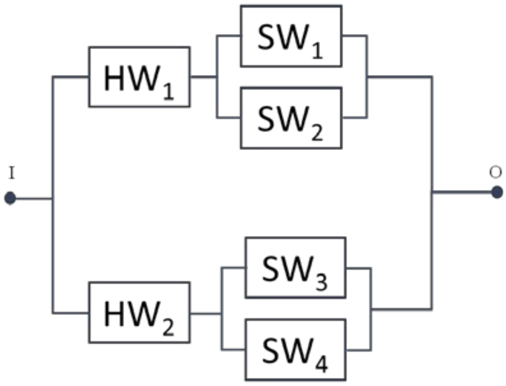

Fig. 3 Reliability block diagram of a duo-duplex architecture

The system in itself corresponds to a simple parallel system, e.g. from two control units. Each of these control units has two software units for function fulfillment. In order to be able to view these software units in the calculation, they are shown in a series connection with the hardware components from a functional point of view, as both the hardware and the software must be functional in order to function properly. Since two software blocks are active on each hardware, they are displayed as a parallel circuit behind the hardware, creating a duo-duplex system. The associated state diagram of the system is shown in Fig. 4.

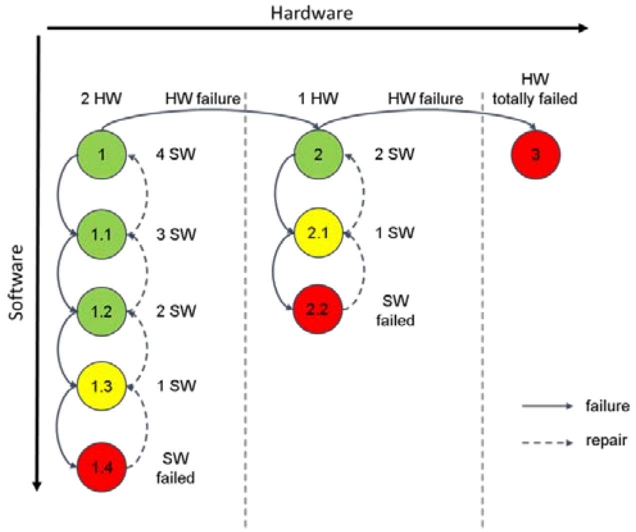

Fig. 4 State diagram of a hard- and software architecture

The state-diagram differs between irreparable hardware failures (horizontal direction) and reparable software failures and repairs (vertical direction). This means that should the system be in state 1.4 (system failed) due to four software errors, it may work again through restarting a software. If the system is in state 3 due to two hardware failures, the system has failed completely and must be fixed, e.g. in a workshop.

For every transition from one state to another special rules can be implemented, for example different kinds of parallel failure and repair modes, depending transitions (Fig. 5) or superordinate rules.

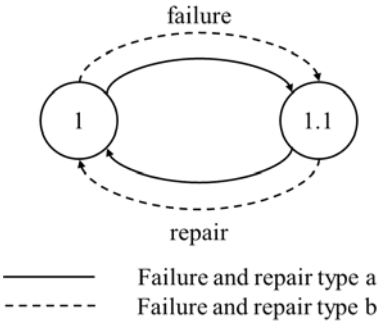

Fig. 5 Parallel and dependent failure and repair modes

Compared to analytical procedures, e.g. Markov processes, much more flexible modelling is possible here. For example, the limitation to the exponential distribution can be bypassed by the simulation so that distribution functions such as the Weibull distribution or lognormal distribution can be used. Moreover, it is possible to adjust the transition rates so that even a dynamic system can be mapped.

By the means of the state diagrams, it is possible to assign key performance indicators (KPI) to the different states in Fig. 4, for example *safety* (green states), *availability* (green and yellow states) and *reliability* (green and yellow states, until the system reaches a red state, after that the reliability equals zero).

In this paper the KPI's are defined with regards to the explained challenges for the software-based safety-critical systems. The advanced definitions are as follows:

Reliability

The reliability Z(t) in this paper specifies the probability, that the system has not failed. For each iteration a failure time $t_{failure;i}$ is determined. This point in time describes the first occurrence of a failed system state, e.g. state 1.4, 2.2 or 3 in Fig. 4. Initially, each iteration starts with $Z_i(t_0) = 1$. As long as the system has not reached a failed state $Z_i(t < t_{failure;i}) = 1$. As soon as the system reaches a failed state, the reliability is defined as $Z_i(t \geq t_{failure;i}) = 0$. The estimation of the reliability $\hat{Z}(t)$ of the system considering all iterations n is displayed in (1).

$$\hat{Z}(t) \cong P(Z(t) = 1) = \frac{\sum_{i=1}^{n} Z_i(t)}{n} \qquad (1)$$

Availability

The duration that the system spends in available states is calculated to determine the availability. States in which the system isn't in a failed state are considered as available states, e.g. state 1 in Fig. 4. The availability $A_i(t)$ of t for one iteration is the ratio of t that the system spends in available states. The calculation of the availability $A_i(t)$ is in (2)

$$A_i(t) = \frac{t_{a,i}}{t}, \qquad (2)$$

with $t_{a;i}$: duration until t, that the system spends in available states.

The estimation of the availability $\widehat{A}(t)$ for several iterations is the arithmetic mean of all iterations n considering all calculated availabilities $A_1(t)$ till $A_n(t)$ (3).

$$\hat{A}(t) \cong \frac{\sum_{i=1}^{n} A_i(t)}{n}. \qquad (3)$$

Safety

The duration that the system spends in safe states is calculated to determine the safety. Safe states are characterized by the fulfilment of fault tolerance and the Monitor-Control-Principle, i.e. there is at least one redundant component that can compare the result of the own calculation with another component, or which could transfer the system in a safe condition. Thus, in the paper the safety is a subset of the availability. The safety $S_i(t)$ of t for on iteration is the ratio of t that the system spends in safe states. The calculation of the availability $S_i(t)$ is in (4)

$$S_i(t) = \frac{t_{s,i}}{t}, \qquad (4)$$

with $t_{s;i}$: duration until t, that the system spends in safe states. The estimation of the safety $\hat{S}(t)$ for several iterations is the arithmetic mean of all iterations n considering all calculated safeties $S_1(t)$ till $S_n(t)$ (5).

$$\hat{S}(t) \cong \frac{\sum_{i=1}^{n} S_i(t)}{n}. \qquad (5)$$

The repeated simulation generates a large number of status profiles. Fig. 6 shows two simulated exemplary status profiles depending on the available redundancy.

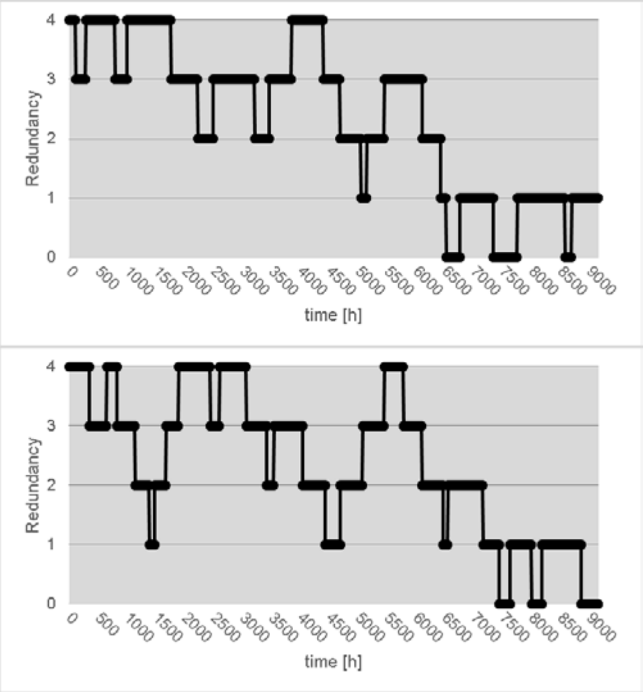

Fig. 6 Status profiles of two iterations

Based on the MCS though the huge amount of status profiles a probability can be calculated that the system is in state X at time t. Combining the probability with the defined state KPI's temporal profiles for safety, availability and reliability can be derived. If different system architectures are examined by means of the MCS, quantitative statements about these systems can be affected. Fig. 7 shows an example of the course of the three KPI's over a time of 12 months.

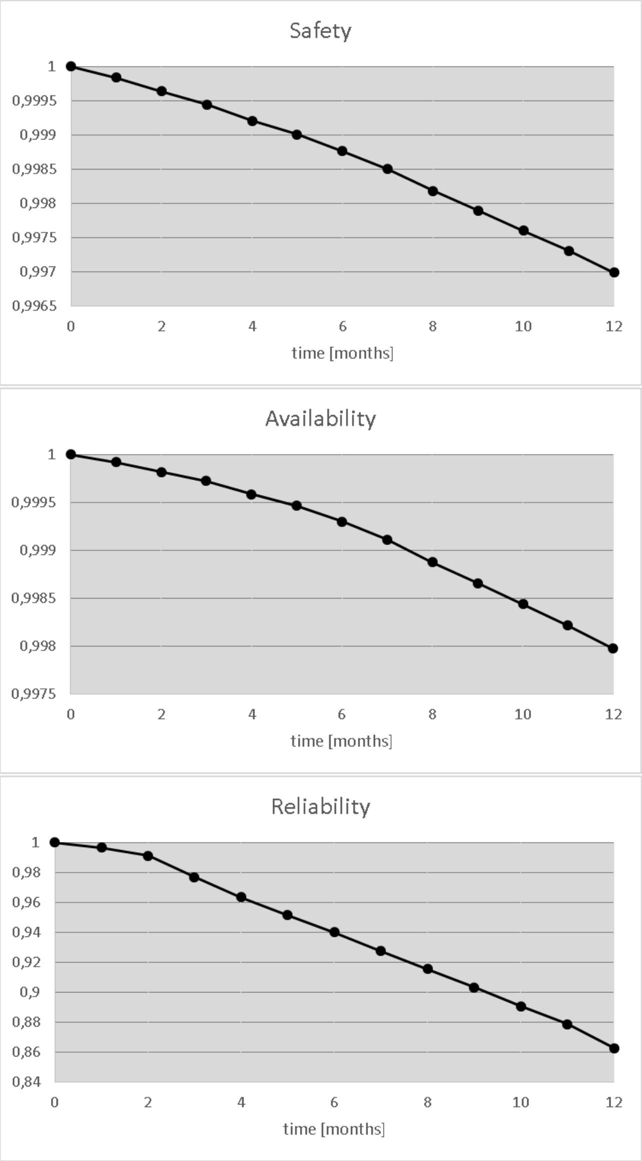

Fig. 7 Diagrams of safety, availability and reliability

As can be seen, the graph of the safety and availability keep a very high level of 99,7% of safety and 99,8% of availability after 12 months. Neglecting a repair after a system failure, the reliability decreases more and is after a period of 12 month at a value of about 86%. The course of safety and availability are very similar. This is because both parameters are based on the state probabilities of states 1, 1.1, 1.2 and 2. In the case of availability, however, the state probabilities of 1.3 and 2.1 are added.

IV. Failure management

An important issue for automated and autonomous systems is to distinguish between systems whose control in the case of an error can be taken back by a person after an acceptance period, e.g. by a driver in the vehicle of Level 3 or via a remote connection by a person in a control center, and systems in which a takeover by a human is not easily possible and the system must continue to operate autonomously in the event of a fault, e.g. space systems that can't be easily controlled remotely due to space communication issues. In the case of the first-mentioned systems, it is sufficient to ensure a temporary functioning until someone else takes control. In systems in which the control can't be transferred to a person in a simple way, continuous functioning in the event of a failure must be ensured with regard to a previously defined fault tolerance. Therefore, an intelligent fault management system is necessary that can identify different types of failures and decide how to handle the failures, either by an independent processing or by a processing via remote.

For space applications, where immediately no person is part of the control loop, the Fault Detection, Isolation and Recovery (FDIR) principle exists [8] [9]. Based on this it is possible to classify faults and to take appropriate measures. As shown in Fig. 8 the classification is split up in levels 0 to 4, in which the criticality of a failure increases from level 0 to level 4.

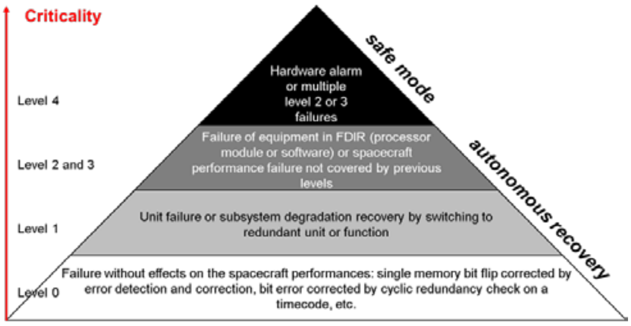
Criticality
Level 4
Level 2 and 3
Level 1
Level 0
Hardware alarm or multiple level 2 or 3 failures
Failure of equipment in FDIR (processor module or software) or spacecraft performance failure not covered by previous levels
Unit failure or subsystem degradation recovery by switching to redundant unit or function
Failure without effects on the spacecraft performances: single memory bit flip corrected by error detection and correction, bit error corrected by cyclic redundancy check on a timecode, etc.
safe mode
autonomous recovery

Fig. 8 Fault diagnosis and management architecture for satellite/spacecraft [9]

If failures of levels 0 to 3 occur, the system can recover itself autonomously. Depending on the severity of the failure, it will be detected and fixed at different levels of the system within variable reaction times. Failures of Level 0 are detected and fixed in each unit of the system by local correction, independent of other units. Level 1 failures must be detected outside the unit and fixed due to a subsystem, which consists out of several units, by switching a unit to its redundant one. If a failure from level 0 or 1 cannot be fixed the failure is categorized in level 2 and has to be fixed more globally at subsystem or platform level. Failures of level 3 are usually software or processor module failures that can be recovered by switching to a redundant processor module. In case of level 4, there are hardware failures or multiple level 2 or 3 failures, the system can no longer handle it autonomously. In case of a satellite it is transferred into a safe mode and the ground station has to intervene and control the system.

This method shows a way how different kinds of failures can lead to different behaviour and that the system can handle most of the failures autonomously by itself, only in very critical scenarios (level 4) a person is needed to deal with the failures and to control the system.

V. Conclusion

This paper presents concepts that can be used to move from today's fail-safe concepts in automobiles to fail-operational systems. In particular, for the introduction of automated driving functions from Level 4 fail-operational systems are inevitable because it can no longer be assumed that a driver is available as a fallback level in the vehicle. For this purpose, redundancy concepts for reliability enhancement were presented, which, in conjunction with the Monitor-Control principle, can also ensure an increase in the safety of the systems. In addition, an approach for the evaluation of software-loaded dynamic system structures was presented. The further development of automated and autonomous technologies automatically leads to new demands in the field of reliability analysis. Classic electric and electronic systems are getting ever more complex functionalities and related deterministic and non-deterministic software building blocks. Generalizations of complex system designs that allow a safety and reliability calculation using classical methods are no longer sufficient. The approach of a state-based Monte Carlo Simulation with extended rules enables a reliability, availability and safety analysis of complex systems. Not so much the individual example is interesting here as the possibilities of modeling, with which a large number of required parameters or other KPIs can be determined. Finally, the FDIR approach was introduced, with which aeronautical and aerospace autonomous fault classification is carried out. Such an approach is not yet available in an automobile, to avoid a check of the system by a remote controller for every kind of failure is the use of such a principle, however, recommended.

The concepts for safe and reliable system architectures presented in this article are only the basis for future activities. In the development of automated and autonomous driving functions the aspects safety and reliability will quickly compete with the question of economy. At this point, it is important to develop well-established and intelligent redundancy structures, which on the one hand meet the normative requirements with regard to safety, e.g. from the field of ISO 26262, but on the other hand also allow it to produce a financially attractive product for the consumer.

The presented simulation model for the safety and reliability proof is currently still in continuous development. The aim here is to be able to map and analyze the systems as detailed as possible in order to ensure a realistic evaluation of the systems. Currently, the systems are modeled manually using a computer algebra software, which makes the mapping and analysis of the systems currently quite costly. In the future, a tool will be available for this, with which the systems can be modeled on one surface and a subsequent evaluation can be carried out automatically.

VI. References

[1] SAE J3016: Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles. SAE International, 15.06.2018.

[2] Rieker, T.: Modellierung der Zuverlässigkeit technischer Systeme mit stochastischen Netzverfahren, Dissertation Universität Stuttgart, 2018.

[3] Raksch, C.: Eine Methode zur optimalen Redundanzallokation im Vorentwurf fehlertoleranter Flugzeugsysteme, Dissertation, Technische Universität Hamburg-Harburg, 2013.

[3] Rehage, D.: Zustandsmodellierung und Zuverlässigkeitsanalyse fehlertoleranter Systemarchitekturen auf Basis von Integrierter Modularer Avionik. Dissertation, Technische Universität Hamburg-Harburg, Schriftenreihe Flugzeug-Systemtechnik Band 1/2009, Shaker Verlag, Aachen ,2009, ISBN: 978-3-8322-8650-7.

[4] Heinrich, J.; Horeis, T.; Plinke, F.: Zustandsbasierte Verfügbarkeitsanalyse von Hard- und Softwarearchitekturen mittels Monte-Carlo-Simulation. Tagung Technische Zuverlässigkeit 2019, Nürtingen, VDI-Berichte 2345, VDI-Verlag GmbH, Düsseldorf, 2019, ISBN: 978-3-18-092345-1.

[5] Plinke, F.: Beitrag zur Weiterentwicklung der zuverlässigkeitstechnischen Sensitivitäts- und Ausfallanalyse mittels Monte-Carlo-Simulation. Dissertation, Bergische Universität Wuppertal, 2015.

[6] Zio, E.: The Monte Carlo Simulation Method for System Reliability and Risk Analysis. Springer Verlag London, 2013, ISBN: 978-1-4471-4587-5.

[7] Meyna, A.; Pauli, B.: Taschenbuch der Zuverlässigkeitstechnik: Quantitative Bewertungsverfahren. 2nd edition, Carl Hanser Verlag, Munich, 2010, ISBN: 978-3-446-41966-7.

[8] Jalilian, S.; Salar Kaleji, F.; Kazimov, T.: Fault Detection, Isolation and Recovery (FDIR) in Satellite Onboard Software. https://ict.az/uploads/konfrans/soft_eng/87.pdf, invoked: 19.07.2019, Azerbaijan, 2017.

[9] Zolghadri, A., Henry, D.; Cieslak, J.; Efimov, D; Goupil, P.: Fault Diagnosis and Fault-Tolerant Control and Guidance for Aerospace Vehicles – From Theory to Application. Springer-Verlag London, 2014, ISBN: 978-1-4471-5312-2.