Towards Integrating Undependable Self-Adaptive Systems in Safety-Critical Environments

Gereon Weiss
Fraunhofer ESK
Munich, Germany
gereon.weiss@esk.fraunhofer.de

Philipp Schleiss
Fraunhofer ESK
Munich, Germany
philipp.schleiss@esk.fraunhofer.de

Daniel Schneider
Fraunhofer IESE
Kaiserslautern, Germany
daniel.schneider@iese.fraunhofer.de

Mario Trapp
Fraunhofer IESE
Kaiserslautern, Germany
mario.trapp@iese.fraunhofer.de

ABSTRACT

Modern cyber-physical systems (CPS) integrate more and more powerful computing power to master novel applications and adapt to changing situations. A striking example is the recent progression in the automotive market towards autonomous driving. Powerful artificial intelligent algorithms must be executed on high performant parallelized platforms. However, this cannot be employed in a safe way, as the platforms stemming from the consumer electronics (CE) world still lack required dependability and safety mechanisms. In this paper, we present a concept to integrate undependable self-adaptive subsystems into safety-critical environments. For this, we introduce self-adaptation envelopes which manage undependable system parts and integrate within a dependable system. We evaluate our approach by a comprehensive case study of autonomous driving. Thereby, we show that the potential failures of the AUTOSAR Adaptive platform as exemplary undependable system can be handled by our concept. In overall, we outline a way of integrating inherently undependable adaptive systems into safety-critical CPS.

KEYWORDS

Self-Adaptive, Cyber-Physical Systems, Autonomous Driving, AUTOSAR Adaptive

ACM Reference Format:
Gereon Weiss, Philipp Schleiss, Daniel Schneider, and Mario Trapp. 2018. Towards Integrating Undependable Self-Adaptive Systems in Safety-Critical Environments. In *SEAMS '18: SEAMS '18: 13th International Symposium on Software Engineering for Adaptive and Self-Managing Systems , May 28–29, 2018, Gothenburg, Sweden.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3194133.3194157

SEAMS '18, May 28–29, 2018, Gothenburg, Sweden

ACM ISBN 978-1-4503-5715-9/18/05.
https://doi.org/10.1145/3194133.3194157

1 INTRODUCTION

In the recent time, advances in artificial intelligence and parallel computing enable novel solutions of CPS in various application areas [17] [12]. A striking example is autonomous driving which has been promoted as next milestone of the automotive industry. These systems are required to reliably function in varying scenarios and react adequately to changes of the environment [18], thus providing a prominent example for the increasing self-adaptation in CPS. For achieving an adaptation and human-like behavior in challenging situations, artificial intelligence and other complex techniques are implemented in deeply embedded systems, which exhibit high demands on performance. Since these systems have a strong impact on the safety of the overall system, they must also incorporate measures to meet the safety goals [15]. In today's prototypes, e.g., automated driving vehicles, the focus is mostly on key features like driving behavior and the capability to work in various scenarios. To enable such systems for the end-user market, high-performant platforms are required which are already available in the CE domain, like graphics processing units (GPUs). However, they inherently lack dependability capabilities for most safety-critical applications. This prevents the realization of novel autonomous systems, which could be achieved by exploiting such powerful self-adaptive CPS. Even though similar advances can be seen in several domains (e.g., collaborative robotics), this work focuses on the application field of autonomous vehicles.

In the following, we present our approach of *Self-Adaptation Envelopes (SA-Es)*, which pave a way towards integrating undependable self-adaptive subsystems into safety-critical environments. They decouple undependable from dependable system parts and provide means to handle respective failures [22]. Through this, unreliable algorithms and platforms can be employed in safety-critical systems. Specifically, we show how self-adaptation is used as a safety mechanism to enable the integration of undependable subsystems. This points out how untrusted platforms (e.g. CE), can be applied in dependable CPS. Moreover, we demonstrate the applicability of this approach in an automotive case study and evaluate its capabilities to handle failures.

The paper is structured as follows. In Section 2 we describe related work to our approach, before we outline the challenge of providing dependability in upcoming autonomous CPS in Section 3. Subsequently, in Section 4 we introduce our concept of SA-Es to

integrate undependable self-adaptive systems into safety-critical environments. We apply and evaluate the approach in an extensive automotive case study and discuss the results in Section 5, before we conclude this work in Section 6.

2 RELATED WORK

Main challenges have been identified by the researching community for providing assurance in self-adaptive systems [6]. They include perpetual assurances, composition and decomposition of assurances, and control theory assurances to handle the uncertainty [8] [25] that inherently arises with these systems. Moreover, using models at runtime [5] enables an assurance at runtime. Such dynamic assurance approaches [4] allow to adjust and alter the system during operation, while ensuring the intended functionality and system requirements. This includes shifting the task of validation and verification to runtime [23], for instance by means of modular safety assurance [21]. Mechanisms as used by online testing [11] allow for checking the proper functioning of a system by shifting tasks to the runtime as well. In contrast, this work concentrates on integrating an undependable self-adaptive system into a safety-critical environment by decoupling it through a SA-E. This is especially needed when the self-adaptive system itself does not include any mechanisms to meet the safety requirements of the system to be integrated in. An example is the integration of legacy systems or platforms originating from non-safety-critical domains. The main exemplary target area we investigate is within the automotive domain. In this field, approaches have been developed in the recent years to enable adaptation of traditionally static automotive systems [27] [19]. Moreover, increasing trust in automotive platforms with respect to security and dependability has been researched, e.g., by adding virtualization and isolation [13]. In addition, our approach considers novel high-performant platforms [3] which are designated for the use in connected and autonomous driving [26].

3 DEPENDABILITY CHALLENGE OF AUTONOMOUS CPS

For autonomous systems, resource-consuming and undependable algorithms need to be implemented in CPS, e.g., artificial intelligence in the form of deep learning [16]. However, respective CPS are often applied in critical scenarios, in which safety concerns must be addressed. Popular examples are industrial control systems, collaborative robots, or autonomous vehicles. In these applications complex algorithms are applied to analyze the environment and plan future behavior. For higher level automation, the system must be able to adapt to the environment and react on changes. One drawback is that algorithms presently deemed suitable for this are non-dependable [10] (e.g., neural-network black-boxes). In addition, their timely execution requires high-performant hardware platforms. However, appropriate and cost-efficient hardware is only available without sufficiently implemented safety mechanisms. Platforms with multi-core GPUs for instance are capable of running deep learning algorithms very efficiently by exploiting parallelization. Despite this, there are generally no safety mechanisms integrated, such as, independent watchdog timers or memory isolation. Firstly, this is the case as they originate from the CE domain and secondly, because of the immense calculation speed such mechanisms must keep up with. Since the designated application areas of CPS are in critical environments, such undependable self-adaptive systems cannot be used without extensive and costly modifications.

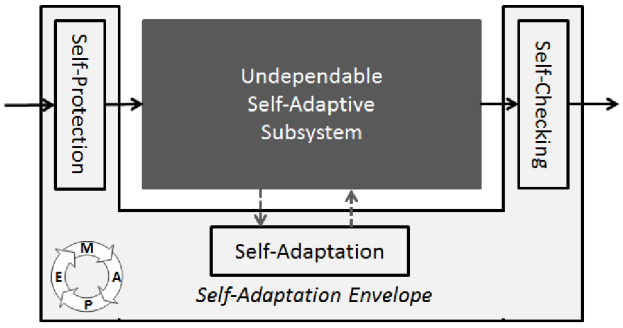

Figure 1: Single SA-E encapsulating an undependable subsystem

When incorporating more autonomy in CPS, also the dependability requirements increase. In the case of autonomous driving (full automation stage), the driver is taken out of the control loop and the car must operate in all situations. Hence, the critical vehicle systems must also work in case of failures, at least until a safe state is reached. This is called *fail-operational behavior*. In order to achieve such a degradation, adequate mechanisms have to be designed and implemented [20]. The individual solutions are often specific through application domain necessities. In low volume, high costs avionics, triple redundancy with voting mechanisms is suitable, whereas in mass products like cars a cost-efficient solution is required. In particular for the latter, integrating less cost-intensive CE could be beneficial. To enable adaptation in these systems, additional measures have to be taken for not endangering the dependability of the overall system. In the case of autonomous movement, if object detection algorithms are adapted to the surrounding, respective deadlines for identifying objects could be missed, potentially leading to a collision. Thus, it must be ensured that adaptations of the undependable system parts do not impact critical parts and keep the system in defined boundaries. The undependable self-adaptive system may be a black-box subsystem to the dependable system. However, the black-box system may be monitored and adapted from the outside, depending on its specificities.

4 SELF-ADAPTATION ENVELOPES

For including an *undependable self-adaptive system (USAS)* in a safety-related system S we introduce a *SA-E*. Its main purpose is to encapsulate the *USAS* and utilize self-adaptation mechanisms to integrate *USAS* into S (cf. Figure 1). It therefore employs a MAPE-K cycle [14] for self-managing *USAS* and decoupling it from its safety-critical environment. The dependable subsystem *DS* is able to degrade to a safe-state, when the *USAS* cannot be relied upon. The SA-E ensures the detection and adaptation of the *USAS*. Therefore, the SA-E checks the input I to *USAS* and suppresses the forwarding if I is out of specification for *USAS* (*self-protection*). The output O of *USAS* is observed as well and cut off, if not valid (*self-checking*). Examples to detect discrepancy of the input are value

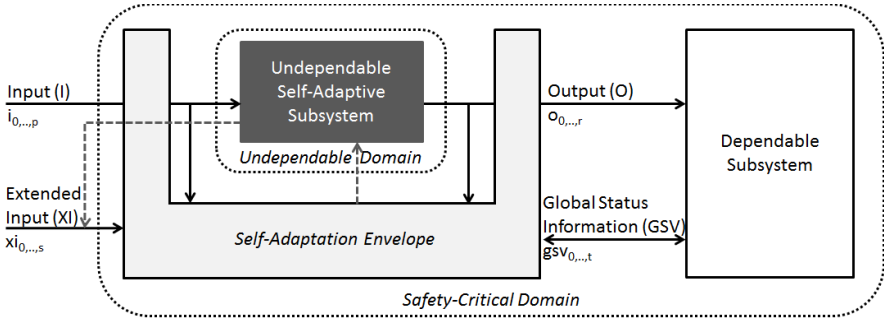

Figure 2: Schematic of integrating an undependable self-adaptive system via a SA-E

boundary checks or cross-validation with single inputs $i_{0,..,p} \in I$ and with other input XI of SA-E. The specific mechanism needs to be individually tailored to $USAS$ and S, just like the SA-E in general. A function $\phi : I \rightarrow \{0, 1\}$ determines if the input is not acceptable by resulting in 0, or 1 if the input is within its specification. The self-checking for instance can be realized by SA-E calculating coarse approximations of the processing of $USAS$ and comparing the value, or by predefined bounds, or assertions for O. The self-checking function $\rho : O \rightarrow \{0, 1\}$ indicates that the output is not okay with a 0 and with a 1 that the output is regarded as being valid.

Furthermore, in case $USAS$ is not a black-box but can instead be monitored and affected by SA-E, additional self-adaptation capabilities can be implemented. If other input can be collected from $USAS$, further diagnosis can be carried out by SA-E. This in turn could, for example, initiate the suppression of the output of $USAS$. An example is the supervision of a reset trigger or flag for memory protection faults. An adaptation of $USAS$ may include reconfiguration or optimization of $USAS$. Examples are de-/activation of services or parameterization of algorithms, depending on the integration depth and options of adaptation provided for SA-E.

An overview of integrating a self-adaptive undependable subsystem in a safety-critical system is depicted in Figure 2. By this approach, the SA-E is for instance capable of handling the following failures (cf. Table 1) and mitigating impact on the dependable subsystem. Overall, the SA-E includes a control loop managing self-adaptation of $USAS$ and safety-related strategies to meet the dependability requirements (cf. Figure 3).

For the monitoring, SA-E observes the input I and output O of $USAS$, additional extended input XI, and status values SV. Based on this, the analysis stage is carried out with respect to specific characteristics of $USAS$. For each characteristic, we introduce a *status value* $sv \in [0, 1]$. It represents the confidence in the correctness of the specific characteristic of $USAS$, where 1 indicates the highest and 0 the lowest confidence. Each SA-E implements $USAS$-specific analysis functions $\alpha_{\mathbf{0},...,\mathbf{n}} : I \times XI \times O \times SV \rightarrow [0, 1]$.

By monitoring and analyzing the input I to $USAS$, extended input XI, the output O from $USAS$, and status values SV (allowing for cascading status values), each α_x calculates the current status value $sv_x, x \in \{0, ..., n\}$. I is the same input as for $USAS$ and O its output. In addition, XI may be additional input to calculate α_x, e.g., coming from S or $USAS$. The latter depends on the available capabilities to monitor additional aspects of $USAS$ beneath its output. An example

Table 1: Exemplary failures handled by SA-E

Failure Type	Example
Failing of $USAS$	reset of $USAS$, detected by monitoring O or XI and omission of output
Corrupted input to $USAS$	sensor failure, detected by min/max I comparison, cross-validation
Corrupted output of $USAS$	AI misinterpretation, detected by boundary check of O
Temporally wrong output $USAS$	missed deadlines because of scheduling problems, detected by monitoring O and expired timer

is, if memory protection is implemented in $USAS$ and its triggering can be monitored as extended input for analyzing the characteristic memory corruption. The above introduced self-protection and self-checking functions of SA-E can be seen as particular realizations of α functions. ϕ calculates a status value which represents the acceptance of $USAS$ input and ρ a status value for the validity of $USAS$ output.

Status values which are relevant for the global system behavior of S are distributed to the dependable subsystem DS, i.e., $SV' = \{sv_0, ..., sv_i\} \subseteq GSV$. Other elements in GSV are status values of characteristics of the DS, which need to be globally considered for the dependability of the overall system.

In the planning and execution phases the GSV is evaluated and a respective pre-defined adaptation plan PAP is chosen. In contrast to $USAS$, the adaptation of SA-E is rather limited. As we are dealing with a safety-critical system, the plans can be pre-calculated and pre-validated in the design. By this, also the consistencies of decentralized adaptations for maintaining the dependability can be ensured. Based on the global system status represented by GSV and according to the adaptation rules AR, a particular adaptation plan $p \in PAP$ is selected. The adaptation rules define which adaptation plan is selected for the present status values. If for example a safety-relevant characteristic's status value is below a defined threshold, a respective adaptation rule will result in planning a

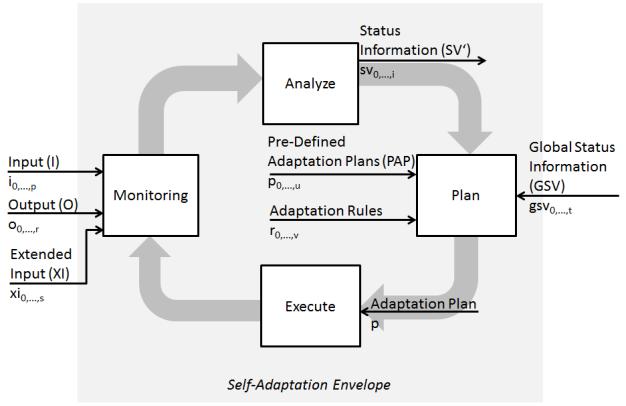

Figure 3: Overview of MAPE-K cycle implemented by SA-E

configuration change. For instance, if the analysis identifies *USAS* output to be out of its specification, SA-E will suppress the output. This is equal to *USAS* transitioning into a fail-silent state, i.e., in case of a failure it will be shutdown. At the same time *DS* will adapt to an emergency operation plan based on the same global system status information. Thus, the *GSV* and adaptation rules *AR*, together with the pre-calculated adaptation plans, represent the basis of the knowledge part of the MAPE-K cycle with respect to safety-relevant adaptations.

If *USAS* can be adapted by SA-E, these adaptations can be planned more freely, as long as they do not conflict with the safety-critical pre-defined plans. A straightforward example is that *USAS* should not be activated, if it should be de-activated according to the pre-designed plan. Of course *USAS* is inherently able to adapt itself, without interference of the SA-E. Only the results of such adaptations of *USAS* can be observed and handled by SA-E, if safety-critical properties represented by relative status values are violated. Thus, the undependable system may incorporate self-adaptation with a high degree of freedom, while the SA-E utilizes pre-planned adaptation to meet dependability requirements of the system.

5 EVALUATION

For showing the applicability and evaluating our approach, we apply it to a real-world case study of autonomous driving. In this scenario, undependable Adaptive AUTOSAR platforms are used for video processing, object recognition, and triggering of the vehicle's braking. A dependable system, consisting of multiple AUTOSAR Classic platforms, implements the SA-E concept and the access to sensors and actuators. We evaluate how exemplary failures of the undependable self-adaptive system, which could in general be derived from a comprehensive safety analysis, can be handled by our approach.

5.1 Automatic Braking for Autonomous Driving

In this real-world automotive use case, an excerpt of an autonomous driving system is used. It shows the part of an autonomous driving system in which braking is activated depending on detected objects in the driving trajectory. Thereby, collisions should be avoided when no evasive maneuver is possible.

In the automotive domain, AUTOSAR represents a worldwide adopted standard for automotive software architectures and a corresponding operating system [7]. It defines a so called *AUTOSAR Classic* platform for traditional automotive systems, from interior control over driver assistance systems to powertrain systems with driving control. In order to cope with the upcoming requirements of autonomous driving, a novel platform is under development which is capable to provide required high-performant computing. This so-called *AUTOSAR Adaptive* platform [9] is mainly service-oriented and able to dynamically adapt its services. Moreover, it is designated to run non-dependable deep learning algorithms for autonomous driving. For these reasons, the adaptive platform can be seen as an undependable subsystem which is integrated into the dependable system of classic platforms. The latter are already being implemented for safety-critical tasks like driving control.

The overall system includes software components for the automated braking and respective hardware components. Figure 4 shows a schematic architecture of the main components, leaving out specific implementation details for the sake of clarity. The general application captures video from the camera system (*Video Capturing*), processes the stream (*Video Processing*), and runs an object detection (*Object Detection*). If a potential collision with an object is anticipated (*Enhanced Brake Assist*), the braking is activated (*Brake*).

In addition, an emergency brake system is used as fallback solution. It is based on radar input (*Radar*) and is not able to differentiate objects as precise as the camera-based system. Therefore, it is implemented as safety-critical dependable system with deterministic algorithms detecting objects in the driving direction. In case the radar data is evaluated and a future collision is determined, an emergency braking (*Emergency Brake Assist*) is issued. However, in comparison to the enhanced braking system only simple situations can be analyzed. From a quality performance perspective, the enhanced braking assistance is preferred. For this reason, if the enhanced braking is reliably working it should be used. As its implementation is not dependable because of the employed technology, the emergency braking should be activated, when the enhanced braking is not working correctly.

The dependable system is implemented with AUTOSAR classic platforms (*ACS*, *ACR* and *ACB*), which execute pre-planned scheduling and meet functional safety requirements like Automotive Safety Integrity Levels (ASILs)[1]. Since the enhanced braking utilizes deep learning algorithms for object detection and video processing, its software compontents must be run on high-performant platforms. Thus, AUTOSAR adaptive platforms *AA1* and *AA2* (two for legacy / implementation reasons) execute the respective software components. They represent an undependable subsystem which is in turn integrated within the dependable braking system. An automotive-specific switched Ethernet network interconnects the adaptive platforms and classic platform via the automotive service-based protocol *SOME/IP (Scalable service-Oriented MiddlewarE over IP)* [2]. The classic platforms communicate through a *CAN (Controller Area Network)* bus. Abstracted data-flows between the software components can be derived from the dashed directed lines in Figure 2. The system is implemented on MBT-2210 MinnowBoards running Embedded Linux with real-time patches and enhancements

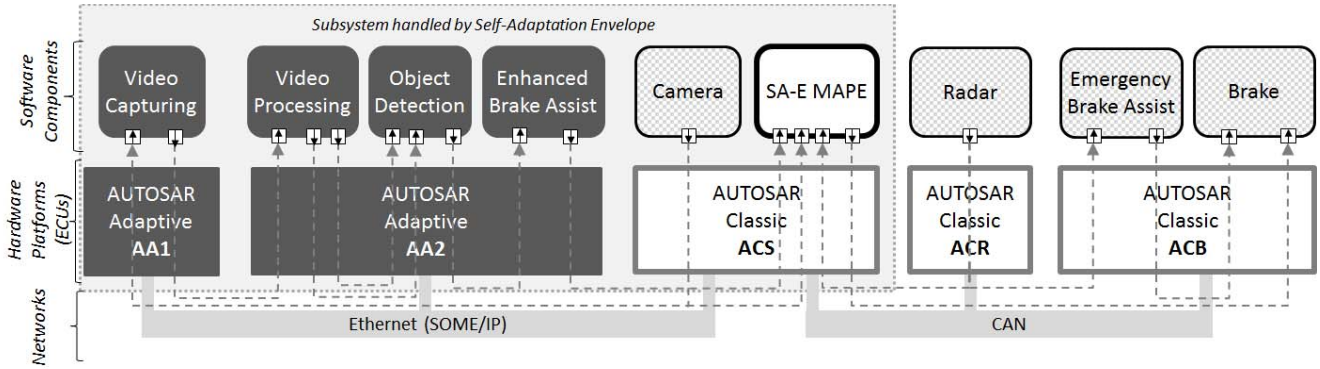

Figure 4: Automotive use case with AUTOSAR Adaptive platforms

for the AUTOSAR adaptive platform. As mentioned, Ethernet with SOME/IP is used for interconnecting the adaptive platforms, and CAN for networking the AUTOSAR classic platforms.

5.2 Applying the Self-Adaptation Envelope

In order to enable integrating the undependable adaptive platforms with the dependable system of classic platforms, we apply our SA-E approach. The main goal is to meet the dependability requirements of the safety-critical braking, although we install undependable adaptive platforms with software components. For a ready product, a safety analysis would identify the failures which must be handled by SA-E. As a comprehensive safety analysis is out of scope of this work, we focus on the main failures which must be handled and how the particular SA-E implements respective mechanisms.

Analogous to Table 1, the following failures are identified and their handling is evaluated:

- **F1**: *Failing of one of the adaptive platforms*
- **F2**: *Corrupted video input from camera to adaptive platforms*
- **F3**: *Corrupted output of the adaptive platforms*
- **F4**: *Too late signaling of braking by the adaptive platforms*

First, we outline the application of the SA-E approach. The SA-E is implemented by a dependable classic platform *ACS* which forwards the camera input to the adaptive platform *AA1* and the braking signal to the classic platform *ACB*. An AUTOSAR software component (SW-C) embodies the MAPE-K cycle of the SA-E (cf. Section 4). In this case the adaptive platforms cannot be self-adapted from the outside by the SA-E. The monitoring stage includes getting the input I, i.e., the camera data, to the subsystem constituted by the adaptive platforms. In addition, it utilizes extended input XI, which is the radar data, and the output of the subsystem, i.e., the braking signal of the enhanced brake assist of *AA2*. In the subsequent analysis stage, status values are calculated by analysis functions based on the monitored data. For simplicity reasons and as it is sufficient in this scenario, we utilize binary status values, which either equal zero indicating invalidity, and 1 in the case of normal operation. In detail, we use status values for the validity of the camera input sv_{Camera} and the enhanced brake output $sv_{EnhBrake}$. The global status value consists of the status of the adaptive platforms gsv_{APs} and of the emergency brake executed on ACB gsv_{EmBA}. In this use case, these values are sufficient to determine the current global system state for handling the dependability of the system.

An analysis function α_{Camera} determines the validity of the camera output and thus, sv_{Camera}. It also implements the self-protection (ϕ), as far as needed for this system. That is, the validity is checked but not intercepted for *AA1*. Instead, the status values of sv_{Camera} and gsv_{APs} are set to zero, indicating that the undependable subsystem is not operating within specification. This in turn leads to invalidating the output of the enhanced braking and using the emergency brake on *ACB* instead. In this example, α_{Camera} detects the amount of noise in the image of the camera. In order to derive $sv_{EnhBrake}$ and the validity of the output, the analysis function $\alpha_{EnhBrake}$ compares the free space captured by the radar with the result of the enhanced brake assist. By this cross-validation, the self-checking is realized. If the results are not contradicting and the timing requirement on computed input is met, $sv_{EnhBrake}$ is set to one, otherwise zero. In addition to cross-validating the output, timing issues of the enhanced braking can be detected by this, e.g., non-synchronized output or missed deadlines. The status of gsv_{APs} is derived by $sv_{Camera}\ AND\ sv_{EnhBrake}$. If either the input or the output of the undependable subsystem is invalid, the status equals zero. Diagnosis mechanisms on platform *ACB* derive gsv_{EmBA}, representing the availability of the emergency braking. Global status values are periodically exchanged between *ACS* and *ACB*, after the analysis phase is completed. During the planning phase, an adaptation plan is selected based on the present status values. For this scenario, there are only two pre-defined adaptation plans needed. One plan p_{fwd} allows for forwarding the output of the enhanced brake assist to the brake, the second plan p_{inh} inhibits the transmission of this output. In case the input to or output of the adaptive platforms is not valid, the omission of output is selected by a corresponding adaptation rule r_{inh}. It defines to utilize the pre-defined plan p_{inh} which inhibits the output, if $gsv_{APs} = 0$. The execution phase carries out the selected plan, i.e., by simply forwarding or not transmitting the enhanced brake assist's signal to *ACB* via CAN bus. By this, from the dependable system's perspective, the adaptive platforms transit into a fail-silent mode. At the same time, a safety mechanism keeps the classic platforms in consistent and safe configurations according to the *GSV* values. As this mechanism is of minor importance for the SA-E approach,

details can be derived from [24]. In this scenario, the indication of invalid adaptive platforms in *GSV* ($gsv_{APs} = 0$), leads to the case that the brake uses the input of the emergency brake assist — while SA-E ensures the isolation of the undependable adaptive platforms by suppressing their output. During normal operation, the enhanced brake assist's input originating from the adaptive platforms is processed.

5.3 Handling Failures

In the following, we assess the handling of the identified potential failures of the undependable subsystem. The capability for handling other failures (e.g., undefined failures) strongly depends on their effect on the monitored input, i.e., the video and radar input, and the calculated output of *AA2*. For example, if the output is not contradicting the radar input, it could not be detected.

F1 *Failing of one of the adaptive platforms:* In case *AA2* fails, a timer in *SA-E* runs out detecting the missed deadline. This in turn results in the global status value $gsv_{APs} = 0$, which leads to suppressing the output of the adaptive platforms and utilization of the emergency brake assist on the dependable classic platform *ACB* as fallback. If *AA1* fails, the fault will be detected as soon as *AA2* calculates invalid output, leading to $sv_{EnhBrake}$ and gsv_{APs} being set to zero and to the same previously described actions. For this system, the indirect failure recognition and potentially delayed reaction are deemed acceptable. However, if required, additional monitoring by SA-E of *AA1*'s video capturing output through the SOME/IP subscription mechanism could identify the failing of *AA1* directly.

F2 *Corrupted video input from camera to adaptive platforms:* If the camera input is corrupted, the analysis function α_{Camera} detects the invalid input. As described previously, this SA-E implementation does not fully implement self-protection, as it is not required. Nevertheless, the status value sv_{Camera} for the input will be set to zero. This forces gsv_{APs} to zero and having the output of the adaptive platforms suppressed, while the fallback emergency brake is used. The detection of corrupted camera input strongly depends on the capabitlities of the analysis function, i.e., what kind of corruption it is capable to detect. Either way, the radar-based detection is in this use case believed to be more reliable than vision-based camera system, as it is used for cross-validation and as fallback.

F3 *Corrupted output of the adaptive platforms:* Only output of the adaptive platforms to the dependable system parts is the enhanced braking data. This is cross-validated with radar information by the analysis function $\alpha_{EnhBrake}$. In any case, the radar data is more trusted and overrules the camera- and deep learning-based detection of the enhanced brake assist. Thus, contradicting output information leads to the status value $sv_{EnhBrake}$ being zero and thus, initiates above described isolation and fallback to the dependable system.

F4: *Too late signaling of braking by the adaptive platforms:* If processing of the adaptive platforms consumes too much time and the data is transferred late, similar to failure F1 the timer of the SA-E runs out. The specific acceptable time can be derived from the period of the enhanced braking function. By this monitoring, the missed deadline of the computations performed on the adaptive platforms, that is for example video processing and object detection, are detected. Same as before, in the case of F1, $sv_{EnhBrake}$ and the global status value gsv_{APs} are set to zero. Hereafter, the output of *AA2* to *ACB* is suppressed and the emergency brake is employed as fallback.

By this automotive example we could validate the applicability of our SA-E concepts. Moreover, we outlined how exemplary failures of an undependable self-adaptive system can be managed to not endanger the dependability of the safety-critical system.

5.4 Discussions

We have presented our initial concept of SA-E for integrating self-adaptive subsystems, which are inherently undependable, into safety-critical environments. However, towards realizing this in real world scenarios, further exhaustive mechanisms have to be elaborated, since this approach is based on several assumptions. Development costs of the SA-E should be below implementing *USAS* as dependable system. Currently, the self-adaptation is seen as isolated process of the dependable system. It is included in a black-box subsystem, leading to the issue that it can only be decoupled and made fail-silent for the rest of the system. Hence, a *USAS* with very low dependability would lead to a high number of fallback situations. Nevertheless, it seems more favorable to also include self-adaptation of the subsystem in the overall strategy to meet the systems' dependability. For example, in case of deep learning algorithms for autonomous driving, fallbacks for single steps of the process from object detection over interpretation and drive planning could be implemented — instead of shutting down the complete system. By this, also deficiencies in accuracy of sensor data could be overcome. We applied our SA-E approach to a specific automotive use case. It still has to be evaluated if the general concepts are also applicable to other areas in which undependable self-adaptive platforms are integrated. Promising applications could be collaborative robots or industrial control systems.

6 CONCLUSIONS

Upcoming CPS need to dependably operate in varying scenarios and under changing conditions. This leads to the demand of incorporating self-adaptation, which in these cases often requires high-end processing power and non-deterministic algorithms. Prominent example are deep learning algorithms being executed on parallel GPUs, which originate from the consumer electronics domain. As promising as the application of such systems in prototypes are, their lack of dependability is an impediment for realizing smart CPS. Thus, we introduced an approach exploiting so-called self-adaptation envelopes. They decouple undependable self-adaptive systems of safety-critical system parts and manage failures, which may have impact on the dependability of the overall system. In an extensive automotive case study within the field of autonomous driving we successfully showed the applicability of our approach and how it is capable to handle identified failures. Thus, our approach provides a way towards integrating undependable subsystems into safety-critical environments. In future work, we will focus on extending the approach to handle not only black-box subsystems but also consider the self-adaptation in maintaining system-wide dependability. This includes taking into account only partially trustable sensor information.

REFERENCES

[1] ISO 26262. 2011. *ISO 26262 Road vehicles – Functional safety.* ISO, Geneva, Switzerland.

[2] AUTOSAR. 2017. *SOME/IP Protocol Specification, AUTOSAR FO Release 1.3.0.* https://www.autosar.org/fileadmin/user_upload/standards/foundation/1-3/AUTOSAR_PRS_SOMEIPServiceDiscoveryProtocol.pdf

[3] D. Becirbasic, M. Molnar, Z. Lukac, and D. Samardzija. 2017. Video-processing platform for semi-autonomous driving over 5G networks. In *2017 IEEE 7th International Conference on Consumer Electronics - Berlin (ICCE-Berlin).* 42–46. https://doi.org/10.1109/ICCE-Berlin.2017.8210584

[4] R. Calinescu, D. Weyns, S. Gerasimou, M. U. Iftikhar, I. Habli, and T. Kelly. 2017. Engineering Trustworthy Self-Adaptive Software with Dynamic Assurance Cases. *IEEE Transactions on Software Engineering* PP, 99 (2017), 1–1. https://doi.org/10.1109/TSE.2017.2738640

[5] Betty H. C. Cheng et al. 2014. *Using Models at Runtime to Address Assurance for Self-Adaptive Systems.* Springer International Publishing, Cham, 101–136. https://doi.org/10.1007/978-3-319-08915-7_4

[6] Rogério de Lemos et al. 2017. Software Engineering for Self-adaptive Systems: Research Challenges in the Provision of Assurances. In *Software Engineering for Self-Adaptive Systems III*, Rogério de Lemos, David Garlan, Carlo Ghezzi, and Holger Giese (Eds.). Lecture Notes in Computer Science (LNCS), Vol. 9640. Springer.

[7] AUTOSAR development cooperation. 2018. *AUTOSAR.* https://www.autosar.org

[8] Naeem Esfahani and Sam Malek. 2013. *Uncertainty in Self-Adaptive Software Systems.* Springer Berlin Heidelberg, Berlin, Heidelberg, 214–238. https://doi.org/10.1007/978-3-642-35813-5_9

[9] S. Fürst and M. Bechter. 2016. AUTOSAR for Connected and Autonomous Vehicles: The AUTOSAR Adaptive Platform. In *2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshop (DSN-W).* 215–217. https://doi.org/10.1109/DSN-W.2016.24

[10] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. In *3rd International Conference on Learning Representations (ICLR 2015).* https://arxiv.org/abs/1412.6572

[11] Julia Hielscher, Raman Kazhamiakin, Andreas Metzger, and Marco Pistore. 2008. A Framework for Proactive Self-adaptation of Service-Based Applications Based on Online Testing. In *Towards a Service-Based Internet*, Petri Mähönen, Klaus Pohl, and Thierry Priol (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 122–133.

[12] S. M. R. Islam, D. Kwak, M. H. Kabir, M. Hossain, and K. S. Kwak. 2015. The Internet of Things for Health Care: A Comprehensive Survey. *IEEE Access* 3 (2015), 678–708. https://doi.org/10.1109/ACCESS.2015.2437951

[13] Christophe Jouvray, Grégoire Chartier, Nicolas François, Ismael Ripoll, Miguel Masmano, and Alfons Crespo. 2010. Enforcing Trust in Control Automotive Platforms. In *Proceedings of the 1st Workshop on Critical Automotive Applications: Robustness & Safety (CARS '10).* ACM, New York, NY, USA, 43–46. https://doi.org/10.1145/1772643.1772656

[14] J. O. Kephart and D. M. Chess. 2003. The vision of autonomic computing. *Computer* 36, 1 (Jan 2003), 41–50. https://doi.org/10.1109/MC.2003.1160055

[15] P. Koopman and M. Wagner. 2017. Autonomous Vehicle Safety: An Interdisciplinary Challenge. *IEEE Intelligent Transportation Systems Magazine* 9, 1 (Spring 2017), 90–96. https://doi.org/10.1109/MITS.2016.2583491

[16] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436.

[17] Jay Lee, Behrad Bagheri, and Hung-An Kao. 2015. A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems. *Manufacturing Letters* 3 (2015), 18 – 23. https://doi.org/10.1016/j.mfglet.2014.12.001

[18] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling, and S. Thrun. 2011. Towards fully autonomous driving: Systems and algorithms. In *2011 IEEE Intelligent Vehicles Symposium (IV).* 163–168. https://doi.org/10.1109/IVS.2011.5940562

[19] Hélène Martorell, Jean-Charles Fabre, Matthieu Roy, and Régis Valentin. 2014. Improving Adaptiveness of AUTOSAR Embedded Applications. In *Proceedings of the 29th Annual ACM Symposium on Applied Computing (SAC '14).* ACM, New York, NY, USA, 384–390. https://doi.org/10.1145/2554850.2554965

[20] Philipp Schleiss, Christian Drabek, Gereon Weiss, and Bernhard Bauer. 2017. Generic management of availability in fail-operational automotive systems. In *Computer safety, reliability, and security. 36th International Conference, SAFECOMP 2017.* 179–194. http://publica.fraunhofer.de/documents/N-470558.html

[21] Daniel Schneider and Mario Trapp. 2013. Conditional Safety Certification of Open Adaptive Systems. *ACM Trans. Auton. Adapt. Syst.* 8, 2, Article 8 (July 2013), 20 pages. https://doi.org/10.1145/2491465.2491467

[22] Elisabeth A Strunk and John C Knight. 2006. Dependability through assured reconfiguration in embedded system software. *IEEE Transactions on Dependable and Secure Computing* 3, 3 (2006), 172–187.

[23] Gabriel Tamura, Norha M. Villegas, Hausi A. Müller, João Pedro Sousa, Basil Becker, Gabor Karsai, Serge Mankovskii, Mauro Pezzè, Wilhelm Schäfer, Ladan Tahvildari, and Kenny Wong. 2013. *Towards Practical Runtime Verification and Validation of Self-Adaptive Software Systems.* Springer Berlin Heidelberg, Berlin, Heidelberg, 108–132. https://doi.org/10.1007/978-3-642-35813-5_5

[24] Gereon Weiss, Philipp Schleiss, Christian Drabek, Alejandra Ruiz, and Ansgar Radermacher. 2017. *Safe Adaptation for Reliable and Energy-Efficient E/E Architectures.* Springer International Publishing, Cham, 1–18. https://doi.org/10.1007/978-3-319-57445-5_1

[25] J. Whittle, P. Sawyer, N. Bencomo, B. H. C. Cheng, and J. M. Bruel. 2009. RELAX: Incorporating Uncertainty into the Specification of Self-Adaptive Systems. In *2009 17th IEEE International Requirements Engineering Conference.* 79–88. https://doi.org/10.1109/RE.2009.36

[26] Devid Will, Lutz Eckstein, Steven Von Bargen, Tessa Taefi, and Roland Galbas. 2017. State of the art analysis for Connected and Automated Driving within the SCOUT project. In *ITS World Confress 2017, Montreal.*

[27] Marc Zeller, Christian Prehofer, Daniel Krefft, and Gereon Weiss. 2013. Towards Runtime Adaptation in AUTOSAR. *SIGBED Rev.* 10, 4 (Dec. 2013), 17–20. https://doi.org/10.1145/2583687.2583691