Safety in Cooperative Automated Systems

embeddedworld2019
Exhibition&Conference
...it's a smarter world

How to ensure critical system properties despite system and context uncertainties

Daniel Schneider, Rasmus Adler, Patrik Feth, Jan Reich, Tobias Braun
Embedded Systems Quality Assurance
Fraunhofer IESE
Fraunhofer-Platz 1, 67663 Kaiserslautern, Germany
name.surname@iese.fraunhofer.de

***Abstract*—Cooperative Automated Systems enable new kinds of applications and services. Corresponding visions stretch across virtually any domain of embedded systems and it is obvious that there lies a huge potential for economic, ecologic and societal improvements and success. However, to unlock this potential we first need to overcome diverse engineering challenges. Most importantly, we need to be able to ensure safety of such systems. Unfortunately, established safety assurance methods and standards do not live up to this task as they have been designed with closed deterministic systems in mind. This paper structures safety assurance challenges of cooperative automated systems and provides an overview and discussion on corresponding solution approaches.**

Keywords—safety; automated systems; autonomous systems; cooperative systems; systems of systems;

I. INTRODUCTION

There is an overarching trend in the application domains of embedded systems towards ever higher levels of automation and interconnection. Both these trends are actually tightly interrelated, because automation can benefit greatly from interconnection and thus cooperation between systems. Through cooperation, the perception scope of constituent systems can be augmented and the perception performance and quality can be improved. At the same time, collaboration allows rendering applications and services, which could not be rendered by single systems alone. As examples, consider the orchestration of agricultural machines in a harvesting operation or the optimization of traffic flow through an intelligent traffic light assistant.

Clearly, such cooperative automated systems harbor enormous potential regarding new types of services and applications. However, before compelling visions and ideas can be turned into actual economic and societal success, we still need to tackle a series of important engineering challenges. One key challenge is the assurance of safety, because established methods and standards operate on the base assumption that systems and relevant system contexts are known and analyzable completely at development time. This base assumption does no longer hold.

On the one hand, in highly automated systems, behaviors are becoming more and more complex, might in parts even be AI-based, and system context and its perception play an increasingly important role. This strongly complicates safety related analysis and argumentation at development time, because there are uncertainties and unknowns which are hard to tackle. Ultimately, based on traditional approaches alone, this leads to worst case assumptions and thus less than optimal system performance.

On the other hand, the aspect of cooperation and corresponding dynamic integration of systems adds additional challenges. Since safety properties of collaborating systems and consumed 3rd party services might not be known, cooperation will be constrained or even not be utilized at all. This again puts a limiting factor on the huge potential which comprehensive cooperation is offering. Additionally, the growing interaction and collaboration introduces (externally) accessible interfaces. These interfaces increase the number of potential attack vectors, which might be abused by malicious attackers. Therefore, safety of cooperative systems is tightly related to security and the impacts between security and safety have to be considered carefully during design. Thus, also the integration of safety and security engineering is another challenge needed to be solved to develop safe cooperative automated systems. This paper sets out to structure the challenges in safety assurance of cooperative automated systems and briefly presents and discusses respective solution approaches. First, in Section II the applicability of current standards as well as open gaps are discussed. In particular, it is elaborated that merely considering functional safety is not sufficient. As additional dimensions, safety of the intended functionality (SOTIF) as well as the engineering of a safe nominal behavior must be considered. In Section III, the paper goes on to discuss how to deal with complex automation behavior and utilization of AI behaviors to, for instance, realize the perception of the environment. Section IV then shifts the focus on openness and modular runtime safety approaches. Section V finally outlines our integrative vision of dynamic risk management. We conclude in Section VI.

II. LIMITS OF CURRENT STANDARDIZATION

Safety standards have different scopes. The scope defines the focused type of technology (software, hardware, mechanics and so on) and the focused domain (medical, avionic, railway, and so on). Apart from some novel standards like Safety Of The Intended Functionality (SOTIF) [1], the scope says nothing about the focused degree of automation and cooperation. This is simply because all systems were closed and had low degree of automation when the standards have been developed. Accordingly, current safety standards provide insufficient guidance for developing cooperative automated systems. In the following, we investigate which new topics and questions come up when we focus on high automation level and dynamic cooperation.

A. Issues with higher automation levels

(Traditional) non-automated systems support humans to implement a plan or decision that the human has made based on some observations. Accordingly, the human is responsible for monitoring the current situation and deriving safe decisions from his observations. The systems only needs to follow the control commands provided at the human-machine interface. Specifying safe system behavior is relatively simple in this case as complex situation awareness and decision making is not necessary for achieving safety. Thus, safety assurance can focus on the handling of malfunctioning behavior (deviations from the specified system behavior) and assume that safety is achieved if the system behaves as intended by his operator. Accordingly, functional safety which is "the part of the overall safety that depends on a system or equipment operating correctly in response to its inputs" [2] makes up a huge part of the overall safety assurance. Considering the world of standardization, functional safety is really good addressed by IEC 61508 and its domain specific derivations: ISO 26262 for road vehicles, IEC 62304 for medical software, EN 50156 for fire alarms, the series of EN 5012x for railway, IEC 61511 for process industry, IEC 61513 for nuclear power, DO-178B for avionic, ISO 25119 for agriculture and so on.

As illustrated in the left box in the Figure below, safety assurance for automated systems requires however much more than functional safety.

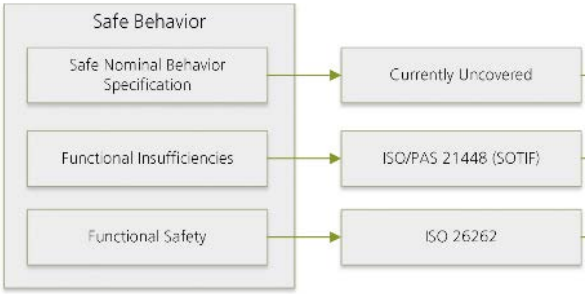

Figure 1. DDI: Aspects of safety and current coverage by standards (here: automotive)

First, it has to be defined what **safe behavior** is. For automated driving, it is for instance necessary to define safe vehicle behavior for all driving scenarios. Some rules like the one for the distance to the vehicle in front might be adopted from existing traffic rules but the existing rules are not sufficient. In special situations, it might be necessary to break the traffic rules in order to avoid an accident. Second, after defining safe behavior, it is necessary to select an appropriate set of sensors and actuators to implement the safe behavior. In this step, one has to deal with **functional insufficiencies** like the insufficiency to detect something with a camera if it is dark. The handling of functional insufficiencies goes also beyond the scope of functional safety. As illustrated in the Figure, recommendations with respect to functional insufficiencies are given by the new SOTIF standard which enhances the functional safety standard ISO 26262. However, standards and/or laws for safe automated driving behavior are still under development. One approach to fill this gap and get a more complete view on the topic of safety is described in [3].

B. Issues with cooperation and collaboration

Another aspect that is not sufficiently addressed by safety standards is the increasing cooperation and collaboration of systems. In a cooperation scenario, systems work together but it is already fixed at design time how they work together. In collaboration scenario, systems negotiate at runtime how they work together. Cooperation is in general the easier case for assuring safety as it is more predictable. But even if it is completely predictable, safety standards provide insufficient guidance. For instance, imagine a back-end server that provides an automated driving vehicle with some information about its environment. If the same information would be provided by a sensor, then ISO 26262 could provide sufficient guidance with respect to software and hardware development. However, as the software and hardware of a backend-sever is totally different, this guidance can hardly be transferred. Standards like ISO 20077 (Extended vehicle (ExVe) methodology), ISO 11783 (Tractors and machinery for agriculture and forestry—Serial control and communications data network) and ISO 11783 (Tractors and machinery for agriculture and forestry—Serial control and communications data network) address some issues of cooperation and collaboration but they do not focus on safety.

To realize cooperative and collaboration scenarios, wireless communication channels have to be integrated either replacing classical bus systems (e.g. CAN, FlexRay, LIN) or connecting single system components or systems (via gateways) to remote components. Regarding the exchange of information, the aforementioned safety related standards focus on faults or effects of faults (e.g. ISO26262-6) and do not consider malicious attackers, who actively try to identify and exploit weaknesses. For the example, a blocked access to the communication channel is considered by ISO26262-6, but not blocking selective exchange information by disturbing the transmissions of selected frames/information, while letting other pass, i.e. an "intelligent" attack. However, even blocking the whole communication is a huge problem, since for wireless communication systems, jamming the communication channels does not require (direct) physical access and might reduce performance or prevent cooperation. Regarding wireless communication, in general the main security requirements for (sensor) wireless networks [4] have to be considered (data confidentiality, availability, data integrity, authenticity and self-organization) to enable a safe and reliable cooperation.

III. ASSURING COMPLEX BEHAVIORS

Systems of higher automation levels are responsible of planning a safe behavior without human assistance. In the automotive domain, this is the case starting from Automation Level 3 [5]. From this level, the driver is allowed to take the eyes off the driving environment. Consequently, the human operator does not constantly monitor the appropriateness of the decisions of the machine controlling the vehicle's motion. From a safety perspective, we see two major challenges with that: First, in Safety Engineering we need to give the possibility to assess the risk of the nominal behavior to come to a decision whether a machine is behaving safely or not. Second, if Machine Learning is used for realizing the behavior, we need to be able to produce Deep Neural Networks of a sufficient level of quality.

A. Risk of the Nominal Behavior

One of the early steps in Safety Engineering is to conduct a risk assessment of the function under development. For hazards caused by component failures, an established process exists to address the requirements of the ISO 26262 standard on the Hazard and Risk Assessment. To claim completeness of the analysis, the process is based on the assessment of worst-case situations for component failures and analyzing the potential consequences in these worst-case situations. Applying the same philosophy to the analysis of nominal behavior would mean to assess the risk of a certain behavior, e.g. a command to steer left, in the worst-case situation, i.e. the situation in which this behavior is least adequate. For any possible behavior, we can easily think of a situation in which this behavior would lead to a very severe accident and where there is a very low chance of avoiding this accident. This means the result of the analysis is pre-determined, which renders the analysis useless. Thus, the philosophy applied for current risk assessment of component failures cannot be transferred to the assessment of the risk of the nominal behavior. To claim completeness of the analysis, i.e. the consideration of a certain behavior in all relevant situations, we thus have to follow a different approach. We anticipate that the assessment of the risk of the nominal behavior will in future equal the assessment of more general, situation-agnostic properties that can be calculated at runtime of the system and consequently in a concrete and actually relevant driving situation. Mobileye is currently advertising such a concept under the term of Responsibility Sensitive Safety (RSS) [6]. We address this issue with Dynamic Risk Assessment as we sketch below. However, we see that more effort needs to be invested to develop a sophisticated and commonly accepted method for the assessment of the risk of the nominal behavior. We currently invest this effort in the SECREDAS project (http://secredas.eu/).

B. Trustworthy Machine Learning

To realize complex system behavior, Machine Learning is playing a major role. Functionalities as the situation perception as well as motion planning are increasingly performed by trained Deep Neural Networks (DNN) as they outperform traditional algorithms. However, there are currently serious quality problems with such DNNs. According to a study presented in [7], machine-learning based systems cause 65% of all disengagements of self-driving vehicles that currently undergo road testing. We see the major reason for that in the lack of an established body of knowledge for the development of high quality DNNs. For the development of software, there exists an established set of best practices. These practices involve inter alia requirements on the development process, as e.g. the use of a certain subset of a programming language and techniques for the review of code or requirements on the testing process. Safety standards as the ISO 26262 contain catalogues of these best practices and give recommendations which methods shall be applied for which level of criticality of the software. However, most of these techniques are specific to software and cannot be transferred directly to the special nature of machine learning. [8] contains a more detailed consideration of this. Even though Deep Neural Networks have been developed for many years now, there is no catalog of best practices available for the development of DNNs. In recent research projects, we are working towards the development of such a catalog to be able to give clear advice to the developers in case a Deep Neural Network shall be applied in a safety-critical context.

IV. ASSURING COOPERATION

The main challenge in ensuring safety of cooperative automated systems (and safety-critical V2X scenarios in general) is to deal with uncertainties and unknowns with respect to the cooperation partners. In other words, one might not know what kind of guarantees come along with a certain information or service of a 3rd party system. Still, it is clearly our aim to utilize such information and services for safety-critical applications, because there is such a huge potential in terms of new applications, improved performance and also improved safety. As an example for the latter, consider systems warning other systems regarding obstacles, systems orchestrating at a crossroad, and so on. Unfortunately, the lack of knowledge regarding external services and their safety properties typically leads to worst-case assumptions, which in turn severely constrain performance, or even lead to the decision not to use external services or information at all.

A straightforward solution to this problem is to enable systems to explicitly negotiate their safety-related properties at runtime. This implies that we establish runtime safety models describing these properties for a (constituent) system and standardize a protocol for their negotiation, thus enabling "just-in-time certification" as it has been envisioned by Rushby [9].

A. Conditional Safety Certificates

Conditional Safety Certificates (ConSerts) [10] are an approach to do exactly that. ConSerts operate on the level of safety requirements. They are specified at development time based on a sound and comprehensive safety argumentation (e.g. an assurance case). They conditionally certify that the associated system will provide specific safety guarantees. Conditions are related to the fulfillment of specific demands regarding the environment what is checked during runtime. In the same way as "static" certificates, ConSerts shall be issued by safety experts, independent organizations, or authorized bodies (depending on the respective application domain) after a

stringent manual check of the safety argument. To this end, it is mandatory to prove all claims regarding the fulfillment of provided safety guarantees by means of suitable evidence and to provide adequate documentation of the overall argument – including the external demands and their implications.

Let us briefly illustrate ConSerts based on the example used in [11]. In the agricultural domain, tractor implement management (TIM) enables implements to assume control over the tractor functions, such as setting the vehicle speed or the steering angle. To do this in the best possible way, the implement might consume sensor information from the tractor of from auxiliary third party sensors, such as a swath scanner or a GPS. Consequently, TIM scenarios are scenarios of cooperative automated systems, realized by cooperation of different systems of different manufacturers.

For the engineering of ConSerts in this example the role of the implement manufacturer shall be assumed. The goal of the manufacturer is to develop a round baler with TIM support. From a functional point of view, it is clear (due to existing standards) how the interfaces between the potential participants look like and how they are to be used. However, the implement manufacturer does not know about the safety properties of these functions.

From a safety point of view, the engineering of the baling application starts top-down with an application-level hazard and risk analysis. Assume that the agricultural manufacturers agreed by convention that during the operation of a TIM application, the application (and thus the application manufacturer) has the responsibility for the overall cooperation. Therefore, the safety engineering goal is to ensure adequate safety not only for the TIM baling application or for the implement, but for the whole cooperation of systems that will be rendering the cooperative application service at runtime. Application-level hazards of the TIM baling application could correspondingly comprise the tractor having an unwanted acceleration or steering during TIM baling. Causes might be located in the TIM baling application itself or in the tractor or in other cooperating systems (e.g. a third party sensor). Causes in the TIM baling application and the implement are tackled by traditional safety engineering. Causes outside the system under development are translated into ConSert demands and runtime evidences that are to be evaluated at runtime. Thanks to the ConSert-based modularization it is thereby sufficient to only consider the direct dependencies of the system under development on its environment. The runtime evaluation can be done bottom up, i.e. the system at the leafs of the cooperation hierarchy determine their guarantees and propagate them up until the root (here: the TIM baling application) can determine its guarantees. Based on these guarantees, the cooperation might be parameterized (e.g. constrain maximum speed) to ensure safety.

The ability to dynamically manage the system performance while always ensuring safety is a strong point of ConSerts. From changes in system guarantees due to wear and tear to changing weather conditions, anything can potentially be considered. Thus, it is no longer necessary to work based on worst case assumptions (because you cannot know the actual conditions during operation), with ConSerts systems become aware regarding the safety-relevant conditions of their environments and can monitor them continuously.

Of course, this flexibility comes at a certain price because additional engineering is required. The engineering and specification of ConSerts and its translation into a machine-readable representation can be a complex task, which should be assisted by adequate tools. By proving an integrated simulations environment, the specification of ConSerts during design time can be supported. Furthermore, simulations enable early testing (fault injections, scenario with complex situations, etc.) and therefore can reduce development costs. Besides this, the results of simulations can also be used as (additional) evidence for the assurance case.

Overall, ConSerts are a relatively lightweight runtime safety approach and they are not far from traditional safety engineering. The main difference being that unknown context is structured into a series of foreseen variants, which are then specified in a runtime model to be resolved at runtime. While this already provides significant gains in terms of flexibility and realizable system performance (compared to a conservative approach), there is still further potential. It is conceivable, that not only a conditional certificate is shifted into runtime but maybe the safety argument (e.g. as dynamic assurance case) itself. In [12], we structure and discuss these options.

B. Digital Dependability Identities

A fundamental problem in dependability engineering is that models exist for many different dependability aspects, which are naturally related through each other by referring to the same system under development. These models are however not related formally with each other so far and thus, no comprehensive reasoning about different system dependability aspects is possible at the same time.

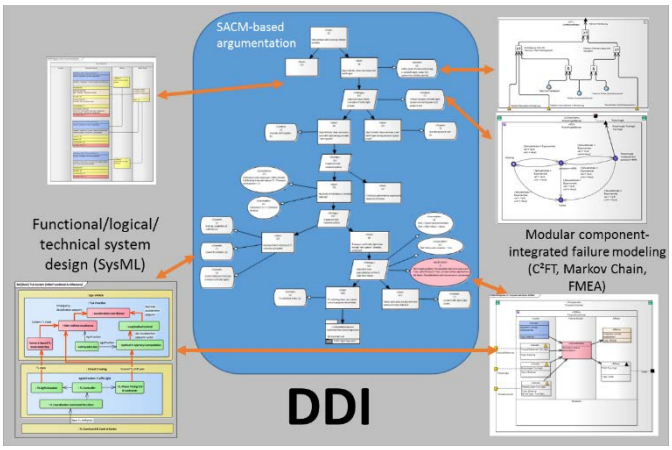

Figure 2. DDI: SACM-based [14] [16] assurance case formally related to its associated dependability models.

The Horizon 2020 DEIS project[1] has the goal to advance the idea of integrating various safety models. As a core concept, the

[1] DEIS: **D**ependability **E**ngineering **I**nnovation for Cyber Physical **S**ystems (CPS); www.deis-project.eu

Digital Dependability (DDI) has been introduced [11] [13]. In general, a Digital Identity is defined as “the data that uniquely describes a person or a thing and contains information about the subject's relationships” [14]. Applying this idea, a DDI contains all the information that uniquely describes the dependability characteristics of a system or component.

A DDI is a living dependability assurance case formally related to all models influencing the satisfaction of a sufficient level of dependability (cf. Figure 2). It contains an expression of dependability requirements for the respective component or system, arguments of how these requirements are met, and evidence in the form of safety analysis artifacts that substantiate arguments. Concretely, these artifacts consist of the system’s or component’s functional behavior, its hazards including risk assessment attributes, its fault propagations in shape of component fault tree and FMEA models, as well as dependability requirements describing the safety concept leading to a sufficient risk reduction and therefore acceptable dependability.

DDIs are produced during design, issued when the component is released, and is then continually maintained over the complete lifetime of a component or system. On the one hand, the exchange format notion of DDIs enables automated engineering support for the synthesis and integration of components into systems during design time considering classical multi-tier supply structures. On the other hand, DDIs explicitly consider the transition from very detailed design time representations into more formal and less detailed DDIs supporting the dynamic integration of systems to "systems of systems" in the field (cf. Figure 3). Runtime DDIs are in a first step based on ConSerts, which have been described in the previous section.

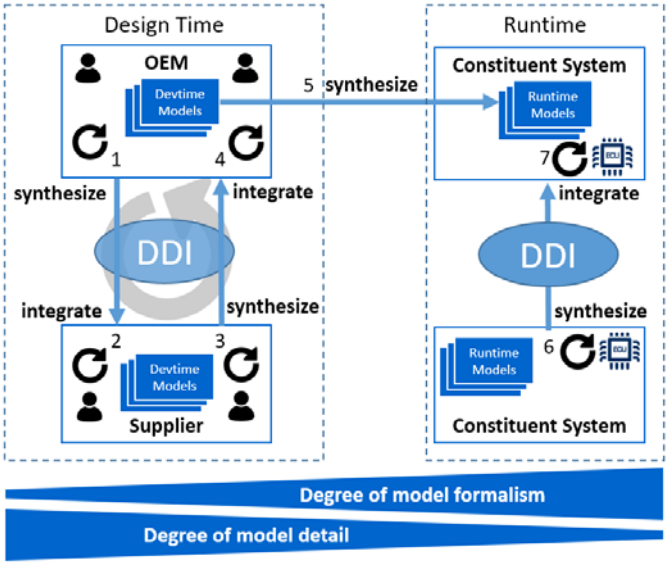

Figure 3. Transitioning from design time engineering automation based on DDIs to runtime dependability reasoning

C. *Importance of Security for Safety*

Given the trend towards more and more interconnection, there are also more and more potential attack vectors to be exploited by malicious attackers. Thus, one could say that without security, a cooperative automated system cannot be considered safe. As a consequence, security must be taken into equation as part of (or at least with clear interfaces to) the safety engineering activities.

As already outlined, interconnections require some kind of communication to exchange information. However, every accessible interface also adds additional potential attack vectors. While safety focus on faults and their effects, security is centered on human-made (malicious) faults [17]. Here, an attacker is considered with a specific motivation, able to adapt in relation to the systems and its vulnerabilities (and even to possible counter-measures) [18]. By successfully gaining access over a system or component, a malicious attacker might also be able to bypass safety counter-measures, since these have been (from the point of view of a safety engineer) designed to address random and systematic faults and not to “compete” with an attacker.

Therefore a holistic development approach combining safety and security engineering activities as well as their implications to each other’s should be the goal. While there are many similarities (e.g. risk as fundamental concept, coverage of the whole life cycle – design and use, related techniques like fault and attack trees [19]), there are also elementary differences such as the assessment of hazards versus threats in form of intelligent attackers [18].

Research projects, such as SECREDAS try to address these issues. SECREDAS is a European project launched in May, 2018 with focus on developing reference architectures, components and verification approaches considering the intertwining of safety, security and also privacy of cooperative automated systems. The outcomes should not only be concepts, but also concrete reusable technology elements and design patterns for their usage (e.g. an extension of Conditional Safety Certificates for unsecure environments as well as a framework for security-safety verification and testing).

V. Towards Dynamic Risk Management

Our dynamic risk management “umbrella-concept” (potentially) integrates many of the approaches outlined in this paper. The concept is illustrated by Figure 2. On the left hand side of the figure, we have different types of relevant environmental context information, which are to be acquired in different ways. There can be sensor-based information provided by the cooperating system of the mission, cloud-based information (such as comprehensive maps or information, i.e. provided by systems not involved in the currently considered mission) and also context acquired by other means such as a direct operator input. Ideally, the context information derived from the environment would be associated with quality and confidence levels to enable more reliable as well as sophisticated analyses and reasoning, enabling a better overall performance. On the right hand side of the figure, the current context of system capabilities is determined (orange arrows). The underlying approach are ConSerts that have been briefly described in Section IV.

Dynamic risk assessment is based on both, the external context information and the internal context of a system. The current capabilities and guarantees of the system are in turn the

basis for determining dynamic risk control measures. They are based on dynamic reconfiguration/self-adaptation of the involved system. The dynamic adaptation has the goal to adapt a system's capabilities to be fit for the current context situation. If this cannot be achieved, the system has to be forced into a fail safe state.

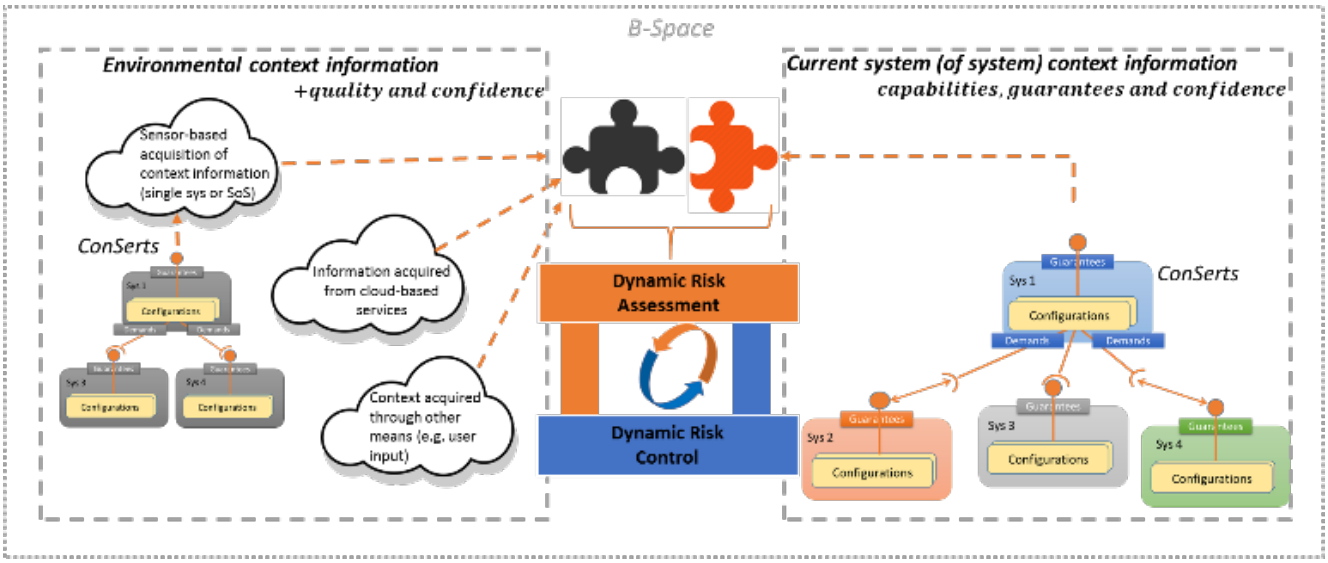

Figure 4. Dynamic Risk Assessment and Dynamic Risk Control

At the same time, while preserving superordinate dependability properties, the mission performance can be continuously optimized. Technically this is achieved by deriving and analyzing the present context and providing the best suitable set of functionality for it. For instance, in the case of a mobile service robot all high-level functionality needed for the present service should be activated while other could be disabled, like maintenance procedures. Such an optimization can take place on different levels depending on its scope. Operational optimization or fail-safe mechanisms will be carried out locally, while strategic optimization is planned on higher abstraction levels and propagated to the single system entities.

Overall, this scheme realizes a typical self-adaptation control loop (cf. MAPE-MART/MAPE-K [20]). Based on context monitoring we identify changes in the environment or the system and conduct dynamic risk assessment (i.e., analysis) to plan and execute corresponding changes (i.e., dynamic risk control). This cycle is running continuously based on runtime models such as ConSerts or DDI. Our vision of comprehensive runtime models and corresponding multi-aspect optimization is further elaborated in [21].

VI. Conclusion

In this paper, we structured fundamental challenges of cooperative automated systems and pointed out respective solution ideas and research directions. We started with a brief analysis and discussion of the related standardization landscape and identified open gaps with respect to the considered class of systems. We then put the limelight on the safety engineering challenges related to complex automation behaviors and the utilization of AI behaviors. Afterwards we put the aspect of cooperation in the foreground and reported on recent work in the direction of runtime safety and dependability models. Finally, we sketched out our vision of comprehensive dynamic management of cooperative systems based on models at runtime.

Even though there are, as we pointed out in this paper, diverse ideas as to how the challenges of cooperative automated systems can be tackled, there are still many open questions. Further research needs to be done and common understanding and guidance needs to be established between all relevant stakeholders (e.g. society, legislation, industry, research) before the full potential of this promising class of systems can be unlocked.

Acknowledgment

The work presented in this paper is supported by the DEIS project – Dependability Engineering Innovation for Automotive CPS. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 732242.

References

[1] ISO PAS 21448, "SOTIF-Safety Of The Intended Functionality", 2019.

[2] https://www.iec.ch/functionalsafety/explained/", accessed at 22.01.2019

[3] Engineering a Safer World: Systems Thinking Applied to Safety (Engineering Systems) (13 January 2012) by Nancy G. Leveson

[4] Adrian Perrig, John Stankovic, David Wagner, "Security in Wireless Sensor Networks" Communications of the ACM, Page53-57, 2004.

[5] SAE International, "Automated Driving Levels of Driving Automation are Defined in New SAE International Standard J3016", 2014.

[6] Shalev-Shwartz, Shai; Shammah, Shaked; Shashua, Amnon, "On a Formal Model of Safe and Scalable Self-driving Cars", Mobileye, 2017.

[7] S. S. Banerjee, S. Jha, J. Cyriac, Z. T. Kalbarczyk and R. K. Iyer, "Hands Off the Wheel in Autonomous Vehicles?: A Systems Perspective on over a Million Miles of Field Data", 2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Luxembourg City, 2018, pp. 586-597.

[8] Czarnecki, K.: “Operational Design Domain for Automated Driving Systems - Taxonomy of Basic Terms”, Waterloo Intelligent Systems Engineering Lab (WISE), 2018.

[9] J. Rushby. “Just-in-Time Certification.” in Proc 12th IEEE International Conference on the Engineering of Complex Computer Systems (ICECCS), Auckland, New Zealand, pp. 15-24, 2007.

[10] D. Schneider, M. Trapp. ”Conditional Safety Certification of Open Adaptive Systems.” ACM Trans. Auton. Adapt. Syst. 8, 2, Article 8, 20 pages, 2013.

[11] D. Schneider et al. “WAP: Digital dependability identities.” In: 2015 IEEE 26th International Symposium on Software Reliability Engineering (ISSRE). Gaithersbury, MD, USA, pp. 324–329.

[12] M. Trapp, D. Schneider. “Safety Assurance of Open Adaptive Systems–A Survey.” Models@run.time, 279-318, Springer International Publishing, 2014.

[13] DEIS Consortium: Engineering Framework for the Generation and INtegratino of Digital Dependability Identities, Whitepaper, http://deis-project.eu/index.php?id=704, accessed: 22-01-2019.

[14] Windley, P., “Digital Identity,” O'Reilly Media, 2005

[15] Object Management Group (OMG): Structured Assurance Case Metamodel (SACM), https://www.omg.org/spec/SACM/About-SACM/, accessed: 17-01-2019.

[16] R.Wei, T.Kelly, R. Hawkins, Model Based System Assurance Using the Structured Assurance Case Metamodel, Journal of Systems and Software, Elsevier, unpublished.

[17] Avizienis A, Laprie J-C, Randell B, Landwehr C., “Basic concepts and taxonomy of dependable and secure computing”, IEEE Transactions on Dependable and Secure Computing 2004;1(1):11–33.

[18] L. Piètre-Cambacédès, M. Bouissou: Cross-fertilization between safety and security engineering, Reliability Engineering & System Safety, Volume 110, Pages 110-126, 2013.

[19] Weiss JD, “A system security engineering process”, Proceedings of the 14th National computer security conference (NCSC), Washington DC, USA, 1991. p. 572–81.

[20] B. Cheng et al., Using models at runtime to address assurance for self-adaptive systems. In Models@ run. time (pp. 101-136). Springer, Cham, 2014.

[21] D. Schneider, M. Trapp, “B-space: dynamic management and assurance of open systems of systems.” Journal of Internet Services and Applications, 9(1), 15, 2018.