PREPRINT: Critical Automotive Applications: Robustness & Safety / CARS@EDCC2022

Continuous Learning Approach to Safety Engineering

Rolf Johansson
Astus AB
Mölndal, Sweden
rolf@astus.se

Philip Koopman
Carnegie Mellon University
Pittsburgh, USA
koopman@cmu.edu

***Abstract*— A phase change moment is upon us as the automotive industry moves from conventional to highly automated vehicle operation, with questions about how to assure safety. Those struggles underscore larger issues with current functional safety standards in terms of a need to strengthen the traceability between required practices and safety outcomes. There are significant open questions regarding both the efficiency and effectiveness of standards-based safety approaches, including whether some engineering practices might be dropped, or whether others must be added to achieve acceptable safety outcomes. We believe that rather than an incremental approach, it is time to rethink how safety standards work. We propose that real-world field feedback for an initially safe deployment should support a DevOps-style continuous learning approach to lifecycle safety. Safety engineering should trace from a safety case to engineering practices to safety outcomes. Such an approach should be incorporated into future safety standards (including ISO 26262) to improve safety engineering efficiency and effectiveness.**

Keywords—continuous assessment, automotive, safety case, continuous integration, integrity level

I. Introduction

Following industry safety standards for Electrical and Electronic (E/E) systems, and especially functional safety standards such as IEC 61508 and its derivatives, is an established approach to creating safe systems. We have a reasonable basis from decades of deployed systems in a variety of domains to conclude that such standards tend to help ensure safety. But we don't know exactly how they accomplish this, nor the degree to which safety outcomes are influenced by the specific activities required by standards vs. a less direct relationship such as merely incentivizing the hiring of safety engineers to be part of a product team. This results in continual pressure to eliminate engineering activities that some feel are superfluous, with such arguments typically based more on intuition or a desire to reduce costs rather than a data driven argument that safety outcomes will not be impaired by skipping steps in safety engineering. Moreover, increasing software content and adoption of novel technologies such as machine learning are dramatically increasing the complexity of deployed safety critical systems. The role of current integrity-level based approaches is in doubt for such future systems without significant changes.

The general approach of E/E system safety has been to require certain process and architectural measures based on expert opinion enshrined in standards. The implicit safety case made is that systems that reasonably follow such requirements will be acceptably safe. Depending on what integrity level is required by operational context and associated risks, different process measures are prescribed in these standards, with more rigorous processes used for higher integrity levels.

Even though these standards are to be updated every few years to track evolving technology, such updates are still based more on expert opinion than on direct, evidenced-based assessments of safety outcomes. Any knowledge base used for such updates is opaque to anyone not on the standards writing committees (and even then some updates are more a matter of opinion and politics than objective engineering data).

Despite decades of use and multiple revisions of key industry safety standards, core questions remain open. How do we know that a certain set of engineering activities and design patterns will achieve a desired level of integrity? And how can we be sure that a given level of integrity will really impart the safety attributes needed for a particular project? (To be sure, we believe that following safety standards is still a best practice and should be done for all safety critical system designs until better practices are identified. The aim of this paper is to encourage the safety community to take steps to further improve the basis and application of such standards.)

We claim that there are significant practical limits to understanding of how safe any particular automotive E/E design is even if a suitable functional safety standard has been followed. There might be a significant number of crashes and incidents attributed to human drivers which could have been avoided or fixed via better system or software design. Currently prescribed measures might be insufficient to guarantee intended safety integrity due to uncertainty as to the predictive power of integrity level engineering practices vs. real-world safety outcomes. There are likely to be some aspects in which current standards are not as rigorous as they should be, resulting in higher than acceptable losses. On the other hand, there might also be a number of process measures prescribed in the standards that are too onerous for ensuring a particular level of integrity compared to alternatives. Some activities required by the standards might even have essentially no effect on safety. We might at the same time be both too loose and too tough in the prescriptions in the standards. Nobody really knows for sure.

Over time, products are constantly becoming significantly more complex, and design methodologies are changing in many ways. These trends further erode any argument that what has worked well for industry in the past is likely to continue to keep working in the future.

The advent of autonomous vehicle technology is injecting a huge discontinuity into automotive safety processes and standards. No longer can automotive safety engineers count on a human driver to take up the slack for loose ends in safety by exercising "controllability." And no longer can the automotive industry "blame" drivers and human error in general for causing crashes or failing to mitigate technical failures by avoiding crashes. This discontinuity will bring to a head the issues and inefficiencies inherent in current safety standards.

Our message is that we must rethink how safety standards work. We start by noting that there has been an implicit safety case for system safety engineers all along. (If nothing else, the safety case is "if we follow this safety standard our product will be safe.") We believe that making the safety case more explicit will form a better basis for a continuous learning approach to understanding why a particular autonomous vehicle is believed to be safe. That more explicit understanding will in turn form the basis for measuring which analysis and architectural patterns are providing how much contribution to safety outcomes – and which engineering rigour techniques required by standards might be omitted with negligible safety effect.

II. Observed safety from the field

For a manually driven road vehicle, safe operation is dependent on both the human driver and on the E/E implemented features controlled by and supporting the human driver. Significant contributing causes to crashes are human driver mistakes, insufficiencies in E/E features, other types of vehicle equipment failures, and road features that are in some way incompatible with road user safety.

A key factor in evaluating E/E contributions to road safety is determining the relative contributions of root causes between E/E issues vs. other sources. Analysing all crashes can be difficult and expensive. More importantly, crash analysis is subject to confirmation bias in which human drivers tend to be blamed for crashes even when there is credible evidence of significant E/E issues that could have contributed to or even be the primary cause of a crash. Such bias toward blaming driver error has been institutionalised and is pervasive [1]. Bias toward blaming driver error can create a large discrepancy between what we think is the integrity of the E/E feature and the real-world safety outcomes.

While general analyses of road vehicle crashes tend to have a driver error bias, some data suggests we might have systematic issues with E/E failures. One such indication comes from comparing software-related safety recalls for passive safety airbag features (SRS) with other software recalls. [2] shows that the frequency of safety recalls for SRS-related software defects is much higher than for software defects with other vehicle features. Moreover, similar failure modes repeat, calling into question the effectiveness of any process learning from previous field failures. If the resulting integrity for all the features following the same standards should be the same, this large difference is hard to understand. And if true it would be hard to defend the applicability of the used standards. However, there are confounding factors likely to be at work here, including uneven application of standards across different companies and different features, and the observation that it is more difficult to blame airbag malfunctions on driver error.

Because of confounding factors in application of standards practices and root cause analysis bias we conclude that we do not have enough evidence today to confidently trace any specific engineering activity according to SIL requirements to safety outcomes. Nonetheless, there is broad experiential data that integrity-based safety standards are generally working in other domains, especially aviation, rail, and industrial controls. So we are left in a situation in which the availability of expert opinion-based engineering standards seems to be yielding safe outcomes – but we are unable to confidently state any causal relationship between the standards and the outcomes.

III. Safety Integrity Levels and their role in a safety case structure

Every safety-critical E/E-related feature has what we will call, for lack of a better term, a *safety story*. That is some sort of narrative to explain to stakeholders why that feature is acceptably safe for its intended purpose. The formality and soundness of the story varies considerably, from a naive unwritten internal monologue for a lone software developer to a detailed written safety case that includes claims, arguments, and evidence or the like. The default safety story for a standards-based engineering approach is that having followed a set of relevant safety standards will necessarily result in safety. The lack of established causal linkages between the contents of functional safety standards and safety outcomes calls any such safety story into question even if best practices are followed.

We consider the minimum acceptable safety story that will be compatible with establishing a causal relationship between engineering practices and safety outcomes to be a safety case that comprises a structured argument, supported by evidence, showing that design and implementation ensures acceptable safety according to domain-relevant safety expectations of stakeholders and real-world use cases. Essentially, we have three aspects to each safety-relevant driver feature: a) determine what being safe implies regarding how sure we need to be (what integrity level) that certain possible failure modes cannot occur, b) distribute the responsibility in the E/E system such that the task of each part of the system element is sufficient to reach overall safety, and c) find a valid argument for each distributed responsibility as to why it can be deemed to have been fulfilled.

Inside most safety standards the prescribed way to connect these three perspectives is by means of Safety Integrity Levels (SILs) or closely related concepts, which are denoted: SIL (ISO/IEC 61508), DAL (ARP 7454 & DO 178), ASIL (ISO 26262), etc. In some cases the approach uses quantitative integrity targets (ISO 21448). The important thing is that these SIL perspectives can be separated and argued independently of each other. Another way to depict the different pieces needed in a safety case is according to the concept of a layered model for structuring safety arguments as outlined in [3]. In this paper we have a focus on the Conformance and Means claims and how they relate to continuous learning, leaving out the aspect of the core rationale claim in the DevOps context, which is elaborated in e.g. [4].

The question of interest to us is how we can know that the engineering techniques required by a standard to attain a particular SIL are (a) actually resulting in the risk mitigation benefits the standard attributes to the SIL, and (b) do not contain fluff activities that are not actually contributing to achieving a SIL in any meaningful way. For now, this is done via the expert opinion of the standards writers.

We would prefer a data-driven approach to associating engineering rigor to safety outcomes. However, we recognize that every E/E system and its use case is different. Moreover, the safety needs for any E/E system change over its lifetime as both risks and operational environments evolve. Therefore we believe that simply using expert opinion to do a one-time update of E/E safety practices associated with any particular standard's approach to SILs will not be enough.

The state-of-the art today is that every safety critical E/E feature should have an associated safety case. A claim regarding safety is supported by an argument and evidence. That argument will in large part be supported by sub-claims that certain engineering approaches have been used, and evidence that the engineering approaches have been applied according to the requirements of a SIL-based standard. Aspects which are not covered by a standard will be argued as might seem appropriate to the design team.

When the safety case includes all necessary safety claims and is thought to be sound (i.e., all claims are believed to be true via adequate supporting arguments and evidence), it is time to release the product – and not before. The important point is that to the degree designers have been following a traditional integrity-based approach the designers merely *think* that the safety case is sound – as far as they know. They have no way to argue the predictive power of their safety case for real world safety outcomes other than experts say following prescribed engineering rigor requirements should be OK.

The industry should admit that legacy safety standards amount to a best practice argument rooted in expert opinion. Rather than blindly trust expert opinion that continually erodes in relevance as technology advances, we should apply best known engineering practices (e.g., existing standards), and then plan to iterate both the system and the standards as we learn more from experience. We propose to do that by wedding safety cases and safety standards to a DevOps approach.

IV. Agile, CI/CD and DevOps

An increasingly important trend in the automotive domain is to continuously deploy new customer features onto existing vehicles rather than releasing a fixed feature set at the start of production (SOP) of a given vehicle model. Applying the agile pattern of CI/CD (Continuous Integration, Continuous Deployment) and DevOps opens opportunities to design for continuous learning. For example [5] proposes a SafeOps concept for continually improving deployed system safety.

CD continually pushes newly validated, released versions of software to road vehicles. Careful "Ops" field engineering feedback data collection can then provide feedback for design and development of future coming versions, i.e. the Dev part of the DevOps. Each release is associated with a complete and self-contained safety case showing that the version is safe via meeting all safety claims. (Significant tooling support will be required for rapid release qualification. Given the need to track changes in the operational design domain and resolve long tail issues discovered only after deployment there is no choice but to do this for autonomous vehicles.) When the deployment has been fulfilled, the next round of Ops data collection starts. In an efficient DevOps loop, there will be a rather high pace of new versions, but with every version having a validated safety case with a predictive power to tell that this release candidate will behave safely when deployed in the road vehicles.

V. Design for Ops Data for Safety integrity

Data collected from each deployment in the DevOps cycle is fed back to improve not only future deployments, but also the quality and completeness of future safety cases. The question is what data to feed back. We must do better than the current recall system under which the number of software defects escapes is increasing quickly [2]. Moreover, we want to apply DevOps and CI/CD in such a way that we can plan for Ops data learning without needing physical vehicle crashes to occur to provide the feedback. It is important that the Ops data is collected at a pace high enough for CI/CD, but yet any crashes are few enough to attain acceptable safety. Note that we do not propose fielding systems that fall short of current safety standard validation requirements. Rather, we acknowledge that we need to do better than we have been at safety engineering continuous learning.

If we use the terminology of the concept of a layered model for structuring safety arguments [3], we can say that the Ops data shall be collected to monitor the conformance claims of pieces of the E/E system. Quantitative conformance claims must be measurable with field data. (In some situations it might be necessary to use quantitative proxies for conformance claims. This is not an inherent limitation of this approach, because any validation accomplished via testing or analysis must necessarily also use a quantitative proxy to determine test pass/fail criteria for the associated conformance claim.)

Whether or not the means claims and environment claims are satisfied is something we determine statically at design time, while Ops data collection is done dynamically at run time. In the ISO/IEC 61508 there is an explicit connection between the SIL attribute values and corresponding quantitative failure rates. In ISO 26262 this bridging is done more implicitly. In any case, if we are to continuously evaluate and calibrate the impact of qualitative measures, we need to be able to connect this to the real observable world of quantitative events. This might be done, for example, via setting project-specific quantitative failure rate targets informed by ASILs for every safety requirement when using ISO 26262.

VI. Ops data and Safety Performance Indicators

Our vision indicates two phases, where the first has a separate focus to make sure that we have a sound base in what is claimed in safety standards as required for each integrity level. We claim that we have a significant knowledge gap there today. In a second phase, emphasis is on continuous improvement of the relevant knowledge in standards, and also collecting Ops datasets for each developing organisation.

In the first phase much of the learning would preferably be done by means of Safety Performance Indicators (SPIs). For the second phase the data collection strategy needs to be more elaborate, because there is no longer a clear-cut falsifiable hypothesis in the form of a product safety case top level claim.

For the first phase, we propose using a specific formulation of a Safety Performance Indicator (SPI) as a quantitative measure for claim satisfaction: *An SPI is a metric supported by evidence that uses a threshold comparison to condition a claim in a safety case* [6]. Any quantitative computations are encapsulated into a threshold comparison, and the result is a logic value related to the truth of the associated claim.

The appeal of this SPI formulation is that it permits instrumenting all types of claims of a safety case and mapping the results to whether the safety argument has been falsified on a claim by claim basis, including not only primary claims, but also sub-claims. If the design team can figure out a way to detect that a claim has been falsified during design time or run time, it can be instrumented with an SPI. Moreover, partial monitors that approximate satisfaction situations can be implemented as SPIs. For example, if a particular measurement approach can only detect some claim falsifications but not all claim falsifications, that can still provide useful feedback regardless of its incomplete nature.

SPIs and supporting claims as they are used in ANSI/UL 4600 not only support safety, but are also used in a defeasible reasoning approach to attempt to defeat claims of safety [7]. This encourages designing SPIs to monitor ways in which a primary claim might be made false by violating assumptions or sub-claims as well as identifying safety case gaps or logic errors. Any SPI threshold violation shows that the safety case is unsound even if vehicle behavior seems safe.

An important benefit of an SPI approach is concentrating feedback on reasons why the product is not as safe as expected rather than on just fixing implementation defects. It is just as important in a safety-critical system to understand how to prevent the next defect via addressing means and environment claims beyond just implementation defects.

VII. Safe Continual Learning

The main goal in the second learning phase is not observing SPI violations to detect safety case falsifications, but rather providing a convincing safety case with predictive power for every version that the deployment is safe. This means that we carefully design our system such that we can achieve a continuous learning and collection in each version, to be used in the safety cases of the following versions. We also extract general knowledge from evidence to integrate learnings across the industry to derive updates of what is considered as needed evidence for each integrity level. Those learnings are used to continually improve safety standards.

There are several ways this can be accomplished, all of which should be used in combination. Each way is described in terms of an operational environment and data collection approach to observing something that could be described as "generalised SPI violations" (the trigger conditions for data collection does not have to indicate a safety case violation).

A first approach is to monitor "generalised SPIs" during design and validation before deployment. This includes monitoring "generalised SPIs" associated with environment and means, as well as conformance and rationale layers during simulation testing and other validation. In this strategy the point of testing is to attempt to trigger "generalised SPI violations" that are a larger set of learning candidate conditions than just to falsify claims in one particular safety case.

Another approach is to operate in some manner of shadow mode. There are several patterns on how to generate the ground truth information for which the shadow mode is evaluated. One example is hybrid approaches such as dependable upgrade applications of the Simplex architecture [8]. Another example is exploiting ASIL B(D) decompositions to find situations in which partial results such as object lists differ between dissimilar channels to indicate a potential design insufficiency. There are many more ways to use shadow mode, with clever design for safety shadow mode Ops data collection being a key part of fast knowledge building.

VIII. Conclusions And Future Work

Debates about the value of engineering activities required by standards yearn for comparisons across different projects to prove (or not) that a specific analysis or engineering techniques provide benefits for safety, code quality, or the like. However, it is almost impossible in practice to arrange such a comparison that produces generally useful process rigor guidance.

Rather than attempting a universal process rigor investigation, we propose a two-phase approach. First, implement requirements for an appropriate integrity level based on available best practices such as standards, then monitor safety outcomes via SPIs to see if acceptable safety has indeed been achieved. Second, monitor the data sources used for SPIs for learning that can be extended to other safety cases and be used to improve safety standards.

The ability of traditional integrity-based approaches to ensure safety has already begun to degrade in the face of increasing system complexity and non-traditional software technology use. Switching to the approach we outline based on continuous deployment of instrumented safety cases with feedback can provide a way to ensure that the right amount of engineering effort is being expended while neither under-shooting nor over-shooting the amount of engineering rigor required to achieve an acceptable level of safety.

References

[1] Koopman, P., "Practical Experience Report: Automotive Safety Practices vs. Accepted Principles," SAFECOMP, Sept. 2018.

[2] Koopman, P., "Automotive software defects," https://betterembsw.blogspot.com/p/potentially-deadly-automotive-software.html accessed June 16, 2022.

[3] Birch, J *et al.*, 'A Layered Model for Structuring Automotive Safety Arguments', in *Proceedings of the Tenth European Dependable Computing Conference (EDCC)*, 2014.

[4] Warg, F *et al.*, 'A Continuous Deployment for Dependable Systems with Continuous Assurance Cases', in *Proceedings of the 2019 IEEE Int. Symp. on Software Reliability Engineering Workshops (ISSREW)*.

[5] Fayollas et al., "SafeOps: a concept of continuous safety," EDCC 2020, pp. 65-69.

[6] Koopman & Kane, A more precise definition of ANSI/UL 4600 Safety Performance Indicators, 2021, https://bit.ly/3HvLsTe accessed June 16, 2022.

[7] ANSI/UL 4600, Evaluation of Autonomous Products, 2nd Edition, Underwriters Laboratories Standards, March 15, 2022.

[8] Sha, A Software Architecture for Dependable and Evolvable Industrial Computing Systems, CMU SEI Tech. Report 95-TR-005, 1995.