Position paper: a vision for the dynamic safety assurance of ML-enabled autonomous driving systems

Alvine Boaye Belle[*], Hadi Hemmati[*], Timothy C. Lethbridge[#]
[*]York University, Toronto, Canada
[#]University of Ottawa, Ottawa, Canada
alvine.belle@lassonde.yorku.ca, hemmati@yorku.ca, timothy.lethbridge@uottawa.ca

Abstract— **Ensuring the progress of autonomous driving technology can save lives, prevent injuries, and enable reductions in traffic volume, accidents, and environmental damage caused by vehicles. Developing industry-wide safety standards and making sure producers of autonomous driving systems (ADSs) comply with them is crucial to foster consumer acceptance. Producers of ADSs can rely on assurance cases to demonstrate to regulatory authorities how they have complied with such standards. Assurance cases are mainly used in safety-critical domains (e.g., automotive, railways, avionics) to deal with high-risk concerns and show to stakeholders that such systems are safe according to domain-specific criteria. Most assurance cases are static i.e., only suitable before the deployment of a system. Dynamic Assurance Cases (DACs) have recently been introduced to provide assurance throughout the lifecycle of a system. However, from our perspective, existing standardized SACs (Static Assurance Cases) notations do not sufficiently support the representation of DACs. This hinders the standardization and adoption of DACs. In this position paper, we propose a novel approach aiming at extending existing standardized SAC notations to dynamically design DACs.**

Keywords— Autonomous driving systems, dynamic safety assurance, machine learning, system assurance and certification.

I. Introduction

An Assurance Case is a "*set of auditable claims, arguments, and evidence created to support the claim that a defined system/service can satisfy particular requirements*" [27]. An Assurance Case is a document that eases the exchange of information between various system stakeholders (e.g., suppliers, acquirers), and between the operator and regulator, where the knowledge regarding a system's requirements (e.g., safety, security, reliability) is convincingly conveyed [24, 27]. In assurance cases, concrete facts such as algorithms, test results, formal reviews, simulations, resource diagrams can serve as evidence relevant to desirable requirements [5, 24]. This evidence is combined with arguments demonstrating how that evidence supports the desirable requirements [5, 24].

Prior to their deployment, systems developed in safety-critical domains require a strong justification that they can effectively support the critical requirements for which they were designed [22]. Thus, it is usually mandatory to develop compelling SACs to support that justification and allow regulatory bodies to certify such systems [12, 29]. This allows verifying the correctness of the systems' capabilities at static time (prior to their deployment) to prevent system failure.

DACs have recently been introduced to provide assurance past a system deployment i.e., at runtime [30]. But according to our analysis, existing standardized SAC notations and semantics do not sufficiently support the various concepts required to represent DACs. This hinders DAC standardization and adoption. Also, relying on existing DAC techniques to assure ADSs (autonomous driving systems) may be challenging because DACs may not be suitable yet to address ADS-specific *epistemic uncertainty* (i.e., *unknown unknowns*).

When using existing approaches, the assurance case developer can neither model nor measure the epistemic uncertainty an ADS may face at runtime. That uncertainty, usually experienced at runtime, is completely unknown and unpredictable, unless it is turned into *aleatory uncertainty* also called "*known unknowns*" [37].Aleatory uncertainty is mostly due to the randomness in a system, such as lost or unobtainable information where the gaps are visible [37].

Most existing assurance case approaches appear not to be suitable for ML (machine learning)-enabled ADSs that operate throughout their lifecycle with epistemic uncertainty. In other words there are unknown unknowns in their risky, dynamic, complex and unpredictable environments, where execution failure may result in loss of life, severe injuries, large-scale environmental damage, property destruction, and major economic loss [3]. Thus, ADSs are likely to be unsafe at runtime if current certification processes are applied.

The goal of our position paper is to show that there is a path towards safer ADSs. Hence, we propose an approach aiming at supporting: 1) higher dynamic safety assurance; 2) hazard elicitation and mitigation; and 3) the mitigation of industrial barriers preventing safety regulation compliance.

II. Background and Related Work

Ensuring the progress of ADSs technology used by autonomous vehicles can save lives, prevent injuries, as well as reduce traffic, costs associated with accidents, and environmental damage caused by vehicles [1, 8].

As explained in [14, 15, 16, 17, 32], machine learning (ML) supports key tasks for autonomous systems (ASs) operations

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

(e.g., obstacle detection, collision avoidance and path planning). ML models are very efficient and effective when provided with input they have already been trained and tested with. But they are considerably less effective at perceiving patterns and predicting outcomes when they encounter unforeseen edge-cases. These may be risky, life-threatening situations such as unprecedented road conditions the ML system has not been exposed to before [1, 38], and could involve myriad issues such as construction, off-road detours, erratic drivers, complex accidents, unusual obstacles (sinkholes, snowdrifts, etc.), the need to follow human signals such as from police officers, or even combinations of these. These unforeseen edge cases embed the epistemic uncertainty an ADS may face at runtime. Thus, the use of ML-enabled components in ADSs must be assured to certify these components are fit for purpose, adequately integrated into their systems, and able to mitigate the edge cases. This is possible by relying on assurance cases.

Assurance cases are a well-established structured technique used to document a reasoned, auditable argument supporting that a system meets some desirable requirements [24]. Assurance cases are structured as a hierarchy of claims, with lower-level claims drawing on concrete evidence, and also serving as evidence to justify claims higher in the hierarchy [13]. Assurance cases are becoming very popular because they provide structured argumentation allowing to communicate safety-critical information [35]. They are mainly used in safety-critical areas (e.g., automotive, nuclear, avionics, healthcare) to deal with high-risk concerns and show stakeholders such systems are safe according to domain-specific criteria [13, 35].

The Goal Structuring Notation (GSN) [19] is the most common graphical notation used to represent assurance cases. GSN diagrams are aligned with the SACM (Structured Assurance Case Metamodel) [27] standard that OMG (Object Management Group) published to promote standardization and interoperability [12]. SACM models the following three major concepts that are further explained in [24]:

1) **A claim**, which is formulated as a proposition and presents a statement that is either *true* or *false*, and which describes what the assurance case is trying to prove.

2) **Some evidence** showing that the claim is reached, constituted from the system's available concrete facts.

3) **A well-structured argument** that links the evidence to the claim in a manner that supports the belief that the evidence supports the claim.

GSN depicts an assurance case as a tree-like structure called *goal structure*. GSN [19] proposes a six-step bottom-up approach to develop goal structures by working from relevant available evidence to the top goal. GSN core elements are goals, solutions, strategies, assumptions, contexts, and justifications [19, 24]. A GSN goal depicts a claim, a GSN strategy depicts an argument and embodies the inference rules that allow inferring a claim from sub-claims [19]. A GSN solution depicts an evidence [19]. GSN assumptions are suppositions about claims [19]. A GSN justification explains why an inference rule or a claim is considered *true* [19]. A GSN context specifies the contextual information needed to interpret a claim [19]. As explained in the GSN specification [19], two types of links allow connecting GSN elements: *SupportedBy* and *InContextOf*. Figure 1 illustrates a partial safety assurance case for UAV (unmanned aerial vehicle) Collision Avoidance that we adapted from [25]. That assurance case is depicted in the GSN.

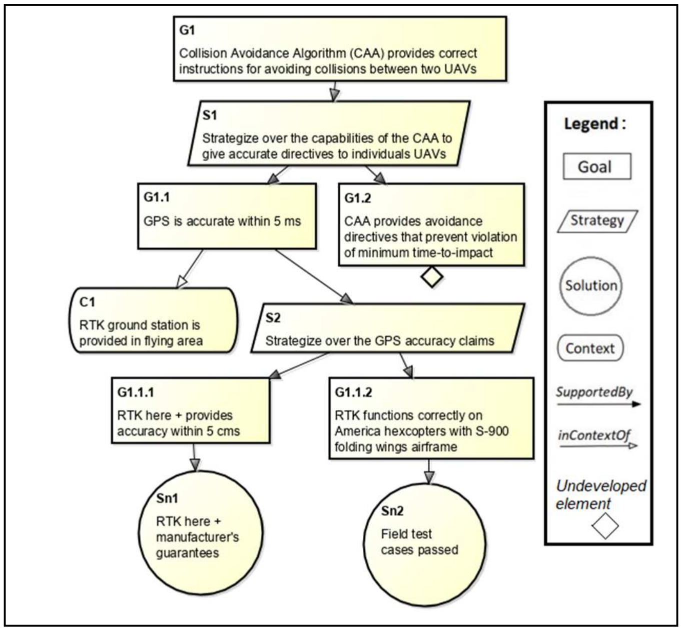
G1
Collision Avoidance Algorithm (CAA) provides correct instructions for avoiding collisions between two UAVs
S1
Strategize over the capabilities of the CAA to give accurate directives to individuals UAVs
G1.1
GPS is accurate within 5 ms
G1.2
CAA provides avoidance directives that prevent violation of minimum time-to-impact
C1
RTK ground station is provided in flying area
S2
Strategize over the GPS accuracy claims
G1.1.1
RTK here + provides accuracy within 5 cms
G1.1.2
RTK functions correctly on America hexcopters with S-900 folding wings airframe
Sn1
RTK here + manufacturer's guarantees
Sn2
Field test cases passed
Legend :
Goal
Strategy
Solution
Context
SupportedBy
inContextOf
Undeveloped element

Fig. 1. partial safety case for UAV Collision Avoidance from [25].

Traditional assurance cases are static i.e., only suitable prior to a system's deployment but not later. We will then refer to them as static assurance cases (SACs). Past the deployment of a system, they may become obsolete, inaccurate and inapplicable [20]. Dynamic assurance cases (DACs) [26, 30] have recently been introduced to assure through-life safety in safety-critical domains (e.g., aerospace) and provide trusted autonomy to autonomous systems (ASs) beyond their deployment [25]. The goal of the DACs is to assure systems throughout their lifecycle. Unlike SACs, DACs have runtime monitors allowing them to continuously assess a system requirements past its deployment [26, 30]. DACs allow supporting operational assessment of assurance and ease intervention and fault-recovery in case change in operational data undermines assurance at run-time [30]. This is crucial for the certification of ADSs since the uncertainty of environments in which they operate will inevitably involve unanticipated risky situations.

Existing techniques (e.g., [19, 21, 23, 30, 36, 38]) for designing assurance cases are usually *ad hoc*, suitable for SACs only, or suitable for DACs but for a limited set of application domains (e.g., avionics, financial and oceanographic domains). These include the study by Hawkins et al. [36] that explains how to use SACs to assure ML components embedded in ADSs, and the approach in [30] that provides a generic framework to dynamically assure ASs but only illustrates the use of DACs to assure safety of ASs in the avionic domain. No design technique nor complementary risk-modeling technique seems suitable to create DACs for ADSs.

Of particular importance, it is our observation that GSN needs to be extended to support the concrete syntax of DACs. Until this is done, DAC adoption and standardization will be hindered, reducing the ability to assure ADSs throughout their lifecycle.

Our work aims at tackling the above issues by exploring the use of DACs to support the dynamic safety assurance of ADSs.

III. Proposed approach

DACs are a relatively new class of certification techniques that can be used to support the continuous assessment and evolution of requirements reasoning, concurrently with system, to provide through-life assurance [30]. We therefore propose to use DACs to provide dynamic safety assurance to ADSs. To increase safety assurance at run time, we need to tackle aleatory uncertainty at design time and epistemic uncertainty at runtime.

Thus, we propose new solutions to design and assess DACs for ADSs to dynamically assure their safety. Our work aims at supporting the representation, and assessment of DACs using an improved version of the very popular GSN controlled vocabulary [12].

We plan to build the an open source tool to support this, on top of the Graphical Modelling Framework. The latter will support the creation of editors based on metamodels defined using Ecore, as provided by Eclipse Modelling Framework. Our tool will be complementary to the ADS. Case studies will focus on ML-enabled ADSs. Figure 2 depicts our proposed approach. The latter consists in three parts that we respectively describe below.

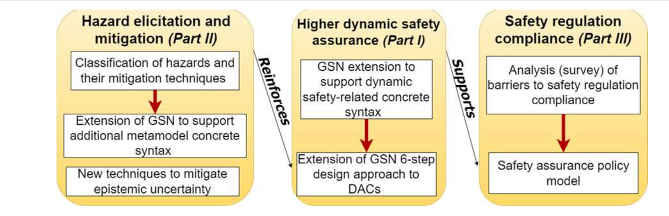
Hazard elicitation and mitigation (Part II)
Classification of hazards and their mitigation techniques
Extension of GSN to support additional metamodel concrete syntax
New techniques to mitigate epistemic uncertainty
Reinforces
Higher dynamic safety assurance (Part I)
GSN extension to support dynamic safety-related concrete syntax
Extension of GSN 6-step design approach to DACs
Supports
Safety regulation compliance (Part III)
Analysis (survey) of barriers to safety regulation compliance
Safety assurance policy model

Fig. 2. High-level view of the proposed approach

A. *Part I: Providing a higher dynamic safety assurance*

We first need to create a taxonomy that identifies and classifies safety-related evidence, as well as confidence and uncertainty assessment techniques for trusted autonomy.

To achieve that, an early tactic will be to use PRISMA 2020 [7], a well-established reporting guideline used in various domains to report systematic reviews (SRs) of existing research and practice and we can focus on ADSs assurance, operation modes, operating contexts. To minimize bias when conducting SRs, we intend to work with at least two researchers who independently analyze search results, and resolve disagreements using common statistical approaches. This will result in an SR that can help create a safety assurance ontology that will facilitate communication [13] among an ADS stakeholders.

Next, we intend to design DACs focusing on safety requirements. These are the most crucial requirements to support in life-critical systems such as ML-enabled ADSs. Supporting the creation of DACs is highly challenging. We propose an innovative approach to do so:

1) *Enrich GSN to support dynamic safety-related concrete syntax* (e.g., monitors, real-time update mechanisms allowing to reconfigure the arguments structures based on the operating contexts). We may do this by analyzing various phases of ML-enabled ADS lifecycle (e.g., design, maintenance, inspection, updates) [32], and taxonomy resulting from our SR;

2) *Extend to DACs the widely used six-step approach that the GSN working group proposed to design SACs* [19]. This can make it possible to propose a design technique able to: 1) Represent through-life monitoring abilities of DACs; 2) Assure a ML-enabled ADS perceives its environment in a way that is suitable for life-critical systems. To create that design technique, we can propose a set of design principles that can provide guidelines on how to iteratively divide a dynamic claim into dynamic subclaims until it can be proved by evidence.

That design technique could allow creating arguments from several argument patterns suitable for the safety of ML-enabled ADSs. These include ML safety requirement assurance argument patterns, ML data argument patterns, ML learning patterns [17, 36]. Runtime monitors can continuously assess the safety of ML-enabled ADS by analyzing evidence from environment and system state [30]. For this purpose, we can explore and adapt the existing runtime monitoring techniques presented in [2]. DACs can use as evidence some of the surveyed categories of evidence. These include perception data collected from ML-enabled ADS's sensors and the data ML-model uses to drive ADS by supporting various tasks such as path planning and motion control. Perception data includes data coming from radar, camera, and lidar sensors. To create dynamic claims/subclaims of DACs at runtime and in bottom-up fashion, we can explore the use of a natural language processing (NLP) techniques [10, 28] exploiting semantic analysis with a natural language generation (NLG) technique [11]. Both can help parse collected evidence, look for their match in the assurance ontology, and support dynamic generation of DAC's claims/sub-claims as statements. We will also explore the use of large language models such as the ones Chen et al. [9] used to generate TGRL (textual syntax of the Goal-oriented Requirement Language) goal models.

The nature of support of each child GSN element to its parent can be specified in the GSN using support patterns [40]. To ensure the consistency in the DACs goal structures, we therefore envision to create instances of such support patterns [40] to properly connect parent dynamic claims to their child subclaims based on the nature of their support to their parent claims. Weaver et al. [40] further describe these support patterns. In particular, the *Single Support Pattern*, corresponds to the situation where a child on its own totally supports the parent goal. With the *Linked Support Pattern*, many child goals interdependently support their parent goals. Finally, with the *Convergent Support Pattern*, many child goals separately support their parent goal.

Approaches used to assess confidence and uncertainty in assurance cases mostly rely on mathematical theories or models (e.g., Dempster-Schafer Theory or Bayesian analysis). These approaches usually allow computing confidence at design-time but may become inappropriate at run-time [4]. To continuously assess DACs at runtime, we can define, as in [31], a confidence measure that adapts one of such approaches (e.g., the approach we introduced in [4]) to the dynamic and stochastic nature of

ADSs. This may result in a probabilistic evidence-based mathematical model that relies on random variables describing the various ADSs states [31]. The assessments made by that confidence measure will be possible by: 1) collecting evidence from ADSs sensors and; 2) evaluating confidence in the safety of the ADS at hand, as well as the uncertainty (risks) stemming from the unpredictability of its current environment.

Epsilon is a scalable, open source, and performant OCL (Object Constraint Language)-like validation language that is compatible with many modeling technologies (e.g., EMF), and whose use involves low overhead [8]. We can continuously validate at run-time well-formedness and completeness of resulting DAC and absence of *known unknowns* in its structure by exploring the use of Epsilon [8] to dynamically generate model validation rules embodying invariants, pre-conditions and post-conditions. This can allow to formally verify the correctness of ADS's capabilities and prevent system failure.

We will explore the use of Epsilon to write model-to-model transformation rules that allow turning a DAC (i.e., an instance of our enriched GSN metamodel) into an EMF-compatible model. We can then explore using EMF to turn models into an Eclipse plugin able to emulate some aspects of human-reasoning when making decisions. This may result in the implementation of a decision-making system in the form of a dynamic fuzzy rule-based expert system using gradient descent-style training, adapted from [21] and [33]. That decision system may automate our new confidence assessment techniques. If assurance level drops below given threshold prescribed by safety regulations, the resulting intelligent system may: 1) instruct ADS to switch to the most appropriate risk-mitigation strategy (e.g., pulling over) given the environment surrounding autonomous vehicle at that specific time; or 2) instruct high-resolution vehicle sensors to look for additional evidence in the current environment surrounding the vehicle to help reprogram the ML model driving the autonomous vehicle to properly handle the identified risks.

In summary, Part 1 of our research will focus on *enhanced notations suitable to represent DACs*.

B. Part II: Hazard elicitation and mitigation to increase dynamic safety assurance

To further enhance the level of safety assurance obtained thanks to Part I, we will proceed with hazard elicitation and mitigation. We will use PRISMA 2020 [7] and interviews with diverse experts (e.g., driving instructors, autonomous driving tech developers, accident investigators) to systematically identify and categorize: 1) every known hazard (i.e., counter-evidence) that may disrupt safe operation of an ADS at run-time; 2) hazard analysis techniques (e.g., functional hazard analysis); and 3) risk mitigation strategies.

We can then propose new risk-based arguments patterns that can help generate at static time claims and counter-evidence able to reason away various categories aleatory uncertainty by:

1) Demonstrating in DAC how it has been mitigated in a system and providing supporting related evidence that falls in surveyed categories of evidence, including evidence related to: engineering rigor (e.g., process quality), functional safety (e.g., detection and shutdown of equipment malfunction), safety of intended function (e.g., resilience to requirements gaps) [1]; or

2) By arguing in DAC such hazards do not affect credibility of overall claim. In this regard, like in [20], we can extend GSN with additional metamodel concrete syntax to support the representation of these new argument patterns together with the assurance deficits they mitigate.

Usually, the assurance case developer can neither model nor measure the epistemic uncertainty (e.g., faults in logical reasoning that the assurance case developer was not even aware of) [37, 38]. That uncertainty is completely unknown and unpredictable, unless it is turned into aleatory uncertainty.

Most existing approaches (e.g., [1, 30, 37, 38]) do not tackle epistemic uncertainty. To address that gap, we will use advanced ML techniques at run-time. For instance, deep reinforcement learning (DRL) can find solutions to a wide range of complex decision-making tasks usually out of reach for traditional ML. It allows learning from experience and can be trained using very few examples instead of a huge number as in traditional ML [34]. These advanced ML techniques can take as input counter-evidence from the ADS's outputs and the data of the ADS's sensors. These techniques aim at dynamically predicting edge cases. By allowing for eliciting some of the unforeseen risks, such techniques may allow turning related epistemic uncertainty into aleatory uncertainty (i.e., known risks). A DAC can then dynamically update its structure by reasoning away the resulting aleatory uncertainty. Data can be sent to the manufacturer: its overall fleet can then be able to deal with the same newly defined aleatory uncertainty. To obtain trustful predictions, we will tackle challenges in training and evolving the target ML models by relying on approaches such as those described in [39]. Our experiments will focus on concrete scenarios (e.g., pedestrian identification) and will aim at demonstrating that our ML techniques can be beneficial to mitigate epistemic uncertainty.

To prevent system failure, we plan to perform runtime verification, simulation and model checking [6]. Well-formedness of the resulting DAC and absence of known unknowns could be checked at run-time by applying Epsilon on EMF model generated from that DAC. Runtime verification and model checking are computationally expensive and may not be suitable for systems like ADSs with real-time requirements. Tackling this challenge will be key to our research.

In summary, Part II of our research will focus on *novel techniques to mitigate hazards at run time*.

C. Part III: Analyzing barriers to safety regulation compliance

Developing industry-wide automotive safety standards and making sure producers of ADSs comply with them is crucial to foster consumer acceptance and trust [1, 8].

Safety regulations include the ANSI/UL 4600 standard for autonomous systems safety (2nd ed.) released in 2022 [1]. This is compatible with existing functional safety standards (e.g., ISO 26262; SOTIF (ISO/PAS 21448)) and aims at ensuring ADSs are safe and reliable for those on road and pedestrians. Still, most companies developing ADSs are reluctant to comply with safety regulations to avoid being liable in case of accidents. This is further compounded by the fear that such regulations could hinder the deployment of autonomous vehicle technologies at scale, thus preventing such technologies from delivering on their potential [29]. To address that, it is crucial to understand industrial barriers to safety regulations compliance. To do so, we will conduct an industrial survey to identify barriers to enforcement of ADSs safety regulations. As in common survey methodologies [18], our survey can consist of series of 5-point Likert and open-ended question types. Participants of the survey can consist of up to 100 diverse companies manufacturing autonomous driving solutions.

The analysis of the survey results should help make recommendations to revise existing safety regulations by making them more aligned with autonomous driving safety constraints. Like [30], we think it is possible to deliver such recommendations in the form of a safety assurance policy model. This is a model-based representation of assurance policies serving as a basis against which the sufficiency of safety assurance can be established by ADSs manufacturers [30].

In summary, part III of our research aims to *make recommendations to enhance safety regulations*.

IV. Conclusion

Hazards caused by autonomous vehicles operated by ADSs are sometimes fatal, which is likely to lead to corporate failure of manufacturers of these vehicles. In this position paper, we therefore propose a novel approach that aims at supporting the dynamic safety assurance of ML-enabled ADSs. Our approach has the potential to create new knowledge and innovative technology to mitigate edge cases at runtime and support more efficiently the dynamic safety assurance of ADSs.

References

[1] Koopman, P. Autonomous vehicles and software safety engineering (May 2022). ICSE keynote.

[2] Guerin, J., Delmas, K., & Guiochet, J. (2022, May). Evaluation of runtime monitoring for UAV emergency landing. In 2022 International Conference on Robotics and Automation (ICRA) (pp. 9703-9709). IEEE

[3] Tambon et al. (2022). How to certify machine learning based safety-critical systems? A systematic literature review. ASE, 29(2), 1-74.

[4] Belle, A. B., Lethbridge, T. C., Kpodjedo, S., Adesina, O. O., & Garzón, M. A. (2019, September). A novel approach to measure confidence and uncertainty in assurance cases. In 2019 IEEE 27th International Requirements Engineering Conference Workshops (REW) (pp. 24-33).

[5] Agrawal et al. (2019, May). Leveraging artifact trees to evolve and reuse safety cases. In 2019 IEEE/ACM 41st ICSE (pp. 1222-1233). IEEE

[6] Calinescu et al. (2017). Engineering trustworthy self-adaptive software with dynamic assurance cases. IEEE TSE, 44(11), 1039-1069

[7] Page et al. (2021). PRISMA 2020 explanation and elaboration. bmj, 372.

[8] Sanchez et al. (2021). Runtime translation of OCL-like statements on Simulink models. Software and Systems Modeling, 20(6), 1889-1918.

[9] Chen, B., et al. (2023). On the Use of GPT-4 for Creating Goal Models: An Exploratory Study. MoDRE workshop at Requirement Engineering conference. To appear .

[10] Wolf et al. (2020, Oct.). Transformers: state-of-the-art natural language processing. In Proceedings of 2020 EMNLP (pp. 38-45).

[11] Dušek et al. (2020). Evaluating the state-of-the-art of end-to-end natural language generation. Computer Speech & Language, 59, 123-156.

[12] Wei et al. (2019). Model based system assurance using the structured assurance case metamodel. JSS, 154, 211-233.

[13] Foster et al. (2021). Integration of formal proof into unified assurance cases with Isabelle/SACM. Formal Aspects of Computing, 33(6), 855-884.

[14] Chen, Y., et al. (2021). GeoSim: Realistic video simulation via geometry-aware composition for self-driving. In Proceedings of the IEEE/CVF CVPR (pp. 7230-7240).

[15] Annable et al. (2022). Model-driven safety of autonomous vehicles. In Recent Trends and Advances in Model Based Systems Engineering (pp. 407-417). Cham: Springer International Publishing.

[16] Karvonen et al. (2020, July). Safety challenges of AI in autonomous systems design–solutions from human factors perspective emphasizing AI awareness. In HCII (pp. 147-160). Springer.

[17] Picardi, C., et al. (2020, February). Assurance argument patterns and processes for machine learning in safety-related systems. In Proceedings of Workshop on Artificial Intelligence Safety (SafeAI 2020) (pp. 23-30).

[18] Badreddin, O., Khandoker, R., Forward, A., & Lethbridge, T. (2021). The evolution of software design practices over a decade: A Long Term Study of Practitioners. J. Object Technol., 20(2), 1-1.

[19] Goal Structuring Notation Working Group. (2021). GSN Community Standard Version 2.

[20] Javed et al. (2021). Towards dynamic safety assurance for Industry 4.0. Journal of Systems Architecture, 114, 101914.

[21] Belle, A. B., & Zhao, Y. (2023). Evidence-based decision-making: On the use of systematicity cases to check the compliance of reviews with reporting guidelines such as PRISMA 2020. ESWA, 217, 119569.

[22] de la Vara et al. (2021). Assurance and certification of cyber–physical systems: The AMASS open source ecosystem. JSS, 171, 110812.

[23] Belle, A. B., Lethbridge, T. C., Garzón, M.,Adesina, O. O. (2018). Design and implementation of distributed expert systems. ESWA, 96, 129-148.

[24] Mansourov, N., and Campara, D. (2010). System assurance. Elsevier.

[25] Vierhauser et al. (2019). Interlocking safety cases for unmanned autonomous systems in shared airspaces. IEEE TSE, 47(5), 899-918.

[26] Denney et al. (2015, May). Dynamic safety cases for through-life safety assurance. In 2015 IEEE/ACM 37th IEEE ICSE (2, pp. 587-590). IEEE.

[27] OMG, Structured Assurance Case Metamodel (SACM), Ver 2.2. June 2021.

[28] Belle, A. B., El Boussaidi, G., & Kpodjedo, S. (2016). Combining lexical and structural information to reconstruct software layers. IST, 74, 1-16

[29] Center for Strategic and International Studies (CSIS). (2021). https://www.csis.org/analysis/driving-future-av-regulations-barriers-large-scale-development. [Accessed in June 2023]

[30] Asaadi et al. Dynamic assurance cases. Computer, 53.12 (2020): 35-46.

[31] Asaadi et al. (2020). Dynamic assurance cases: a pathway to trusted autonomy. Computer, 53(12), 35-46.

[32] Ashmore et al. (2021). Assuring the machine learning life cycle. CSUR, 54(5), 1-39.

[33] Straub, J. (2021). Impact of techniques to rule-based expert system gradient descent networks. Journal reduce error in high error of Intelligent Information Systems, 1-32.. (2021)

[34] Kiran et al. (2021). Deep reinforcement learning for autonomous driving. IEEE Transactions on Intelligent Transportation Systems.

[35] Maksimov et al. (2019). A survey of tool-supported assurance case assessment techniques. ACM Computing Surveys (CSUR), 52(5), 1-34

[36] Hawkins et al. Guidance on the assurance of machine learning in autonomous systems (AMLAS). arXiv:2102.01564, 2021.

[37] Duan et al. (2017). Reasoning about confidence and uncertainty in assurance cases: A survey. In Software Engineering in Health Care. Springer International Publishing, 64–80.

[38] Chechik, M., Salay, R., Viger, T., Kokaly, S., & Rahimi, M. (2019, April). Software assurance in an uncertain world. In FASE (pp. 3-21). Springer.

[39] Dulac-Arnold et al. (2021). Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. Machine Learning, 110(9), 2419-2468.

[40] Weaver et al. (2003, October). A pragmatic approach to reasoning about the assurance of safety arguments. In Proceedings of the 8th Australian workshop on Safety critical systems and software-Volume 33 (pp. 57-67).