# SAFEXPLAIN: Safe and Explainable Critical Embedded Systems Based on AI

Jaume Abella[1], Jon Perez[2], Cristofer Englund[3], Bahram Zonooz[4], Gabriele Giordana[5], Carlo Donzella[6], Francisco J. Cazorla[1], Enrico Mezzetti[1], Isabel Serra[1], Axel Brando[1], Irune Agirre[2], Fernando Eizaguirre[2] Thanh Hai Bui[3], Elahe Arani[4], Fahad Sarfraz[4], Ajay Balasubramaniam[4], Ahmed Badar[4] Ilaria Bloise[5], Lorenzo Feruglio[5], Ilaria Cinelli[5], Davide Brighenti[7], Davide Cunial[7]

[1] Barcelona Supercomputing Center, Spain
[2] Ikerlan Technology Research Centre, Basque Research and Technology Alliance (BRTA), Spain
[3] RISE Research Institutes of Sweden, Sweden
[4] Navinfo Europe, The Netherlands
[5] AIKO s.r.l., Italy
[6] Exida Development s.r.l., Italy
[7] Exida Engineering s.r.l., Italy

*Abstract*—**Deep Learning (DL) techniques are at the heart of most future advanced software functions in Critical Autonomous AI-based Systems (CAIS), where they also represent a major competitive factor. Hence, the economic success of CAIS industries (e.g., automotive, space, railway) depends on their ability to design, implement, qualify, and certify DL-based software products under bounded effort/cost. However, there is a fundamental gap between Functional Safety (FUSA) requirements on CAIS and the nature of DL solutions. This gap stems from the development process of DL libraries and affects high-level safety concepts such as (1) explainability and traceability, (2) suitability for varying safety requirements, (3) FUSA-compliant implementations, and (4) real-time constraints. As a matter of fact, the data-dependent and stochastic nature of DL algorithms clashes with current FUSA practice, which instead builds on deterministic, verifiable, and pass/fail test-based software. The SAFEXPLAIN project tackles these challenges and targets by providing a flexible approach to allow the certification – hence adoption – of DL-based solutions in CAIS building on: (1) DL solutions that provide end-to-end traceability, with specific approaches to explain whether predictions can be trusted and strategies to reach (and prove) correct operation, in accordance to certification standards; (2) alternative and increasingly sophisticated design safety patterns for DL with varying criticality and fault tolerance requirements; (3) DL library implementations that adhere to safety requirements; and (4) computing platform configurations, to regain determinism, and probabilistic timing analyses, to handle the remaining non-determinism.**

## I. Introduction

CAIS industries (e.g., automotive, railway, and space) show an increasing interest in DL-based techniques. This trend is driven by several reasons:

- *Digitization of CAIS*: The number of mechanical subsystems enhanced or completely replaced by electronic components is increasing, Advanced software functions are becoming ubiquitous to control all aspects of CAIS, and safety related systems are not an exception. The digitization of CAIS can bring huge benefits to the society such as (1) safer roads, skies, and airports, preventing 90% of collisions per year [1]; (2) dramatic reduction, up to 80%, of carbon dioxide ($CO_2$) profile of different types of vehicles [1]; and (3) improved quality of life by reducing the time people spend driving or waiting for train and flight delays, and making vehicles accessible to people that would be otherwise excluded for disability or economic reasons.
- *Effectiveness of AI techniques*: AI techniques (and DL in particular) are at the very heart of the realization of advanced software functions such as computer vision for object detection and tracking, path planning, driver-monitoring systems, gesture-activated AI assistants, and voice-based command and control [2]. Hence, AI solutions are a cornerstone in the development of future advanced (fully) autonomous systems. As such, AI is considered crucial for providing functionally-critical features such as perception (obstacle detection), path/trajectory planning, and vehicle tracking in state-of-the-art autonomous vehicles [3], [4].

Autonomous operation is the epitome of safety-related applications of AI in CAIS, and exemplifies the need for increasingly high computing performance whilst making AI solutions to comply with FUSA requirements.

The malfunctioning of a CAIS can lead to unacceptable consequences either just economic or on the environment and people, even causing fatalities. Hence, CAIS are developed according to a set of requirements on the processes and analyses for their functional and non-functional behaviour aiming at keeping the risk of hazardous events below an acceptable level. These requirements are defined by generic (e.g., IEC 61508 [5]) and domain-specific certification standards, like ISO 26262 [6] for automotive and EN5012x [7], [8] for railway. Within the safety process, explainability of software and traceability of requirements are mandatory, either implicitly or explicitly. DL-based advanced software functions are no exception and must comply with the CAIS certification requirements. Complementary standards, such as ISO/PAS 21448 [9] and ANSI/UL 4600 [10], among others, are also being developed to address safety concerns of autonomous systems that use AI. However, these new standards are still in their infancy, and, for instance, ISO/PAS 21448 (aka SOTIF) for automotive focuses on what should be covered during the system engineering, but leaves out how to achieve the goals.

Safety-related development processes build on the unambiguous specification of FUSA requirements and steps like:

1) Deterministic algorithms fulfilling safety requirements,
2) Verifiable implementations of those algorithms,
3) Test campaigns to validate that safety requirements are not violated with pass/fail tests.

However, current practice in DL frontally clashes with these FUSA-related processes since:

1) DL software is built as a combination of control (model configuration, such as what layers to use, in which

TABLE I: Relation of SAFEXPLAIN's main goals to DATE 2023 topics

| Topic | Title | SAFEXPLAIN |
|---|---|---|
| - | Special Days on Emerging Topics. Autonomous Systems Design | SAFEXPLAIN is fully aligned with this Session on the use of AI in CPS like automated driving, and avionics and addresses key challenges in this area related to verification of dependable autonomous systems. |
| E2 | Real-time, Dependable and Privacy-Enhanced Systems | SAFEXPLAIN covers aspects related to software timing validation of AI software in heterogeneous MPSoC platforms including GPUs |
| E3<br>E4<br>E5 | ML Solutions for Embedded and Cyber-Physical Systems<br>Design Methodologies for ML Architectures<br>Design Modelling and Verification for Embedded and CPS | SAFEXPLAIN research focus is on novel methods for the design, development, verification, and deployment on AI-based solutions for embedded critical system that support FUSA certification objectives. |
| D8<br>D9 | Network-on-Chip and on-chip communication<br>Architectural and Microarchitectural Design | Time predictability is one of the main traits at the platform level. SAFEXPLAIN . will identify major predictability walls in the NoC/processor that affect determinism; and will benefit from any support to reduce it |
| T3 | Dependability and System-Level Test | SAFEXPLAIN will contribute with techniques able to manage errors spanning from both, hardware random faults and DL-related mispredictions. Common safety measures should be devised for efficiency reasons. |

order, etc.) and data (algorithm parameters are obtained from training with specific datasets) with a stochastic and data-dependent nature.

2) There is a lack of sufficient explainability and traceability: it is not properly specified why each DL layer is used, its semantics, and why layers are deployed in a specific order (i.e. their composed semantics) so that requirements can be traced end-to-end. Nor it is specified what the scope of application is (e.g. valid input data range), and the confidence that can be reached on the obtained predictions (e.g. by detecting occlusions).
3) Prediction accuracy is stochastic, and test campaigns deliver, in the best case, success rates linked to specific testing datasets, therefore exposing to dataset-dependent test conclusions in many cases.
4) High-performance hardware in which DL solutions executes is poorly predictable, challenging the use of deterministic timing analysis techniques. Also on the implementation side, DL libraries are not designed according to any FUSA standard and make use of features typically discouraged by safety standards (e.g., pointers and dynamic memory) as challenging qualification.

This paper presents SAFEXPLAIN, a newly started Horizon Europe project (October 2022 - September 2025) in its first stages. SAFEXPLAIN leverages the fact that the transition to fully autonomous systems will be incremental, with increasing safety-related functions controlled by AI software until reaching full autonomy (e.g. level 5 for automotive [11]). This creates an evolving and complex environment for certification as safety needs will vary across each increment in autonomy. This results in different AI (DL indeed) usage levels (i.e. safety requirements) of AI software.

SAFEXPLAIN tackles the above challenge by targeting a novel and flexible approach to allow the certification – hence adoption – of DL-based solutions in CAIS by:

- Carefully analyzing the gap between current FUSA development processes and existing hardware and software DL solutions.
- Devising safety patterns for different DL usage levels (i.e. with varying safety requirements) to allow using DL in any CAIS functionality, for varying levels of criticality and fault tolerance. Safety patterns will allow to develop an incremental safety approach that adapts to the needs of every AI (DL) usage level of AI software until reaching autonomous operation.
- Architecting DL solutions that allow explaining why they satisfy FUSA requirements, with end-to-end traceability, with solutions to explain whether predictions can be trusted, and with strategies to reach (and prove) correct operation, in accordance with certification standards.
- Proposing MPSoC configurations, to regain predictability as much as possible, and probabilistic timing analyses, to handle the remaining non-deterministic timing behaviour. This is complemented with DL implementations that favor qualification and timing analysis.

The rest of the paper is organized as follows. Section II shows the relevance of SAFEXPLAIN research for DATE topics. Section III presents an analysis of the challenges for the use of DL software in CAIS. Sections IV and V introduce the SAFEXPLAIN approach and methodology respectively. Section VI summarizes the paper.

## II. Relevance to the DATE Conference

SAFEXPLAIN is aligned with several DATE 2023 topics and special sessions. The main common ground stems from the traction that AI software is getting in Cyber Physical Systems (CPS) that spans safety related challenges to the hardware and software. Table I lists some of the DATE 2023 topics on which SAFEXPLAIN can contribute or take benefit from.

As shown, SAFEXPLAIN relates to the Special Days on Emerging Topics and, in particular, to Autonomous Systems Design that seek for works addressing the challenge on "the design and verification of dependable autonomous systems". It relates in E2 to the software timing-related aspects achieved via smart computing platform configuration and novel probabilistic timing analysis techniques. SAFEXPLAIN can also benefit by any hardware techniques (D8 and D9) in hardware resources like NoCs to regain time determinism. SAFEXPLAIN deals with the safe use of DL in critical autonomous CPS, so it naturally fits under the DL and CPS related topics E3, E4, and E5. Last but not least, the FUSA development process has a strong component of testing during validation phases, and of dependability during architectural design by deploying safety measures to manage different types of faults, hence matching T3 topic.

## III. The Challenge

There is a fundamental gap between FUSA requirements of CAIS and the nature of DL solutions deployed to deliver CAIS functionalities. The lack of explainability and traceability, and the data-dependent and stochastic nature of DL software, clash against the need for deterministic, verifiable and pass/fail test-based software solutions for CAIS. This section provides the first contribution of this paper, consisting in an analysis of the ramifications of the FUSA-DL gap.

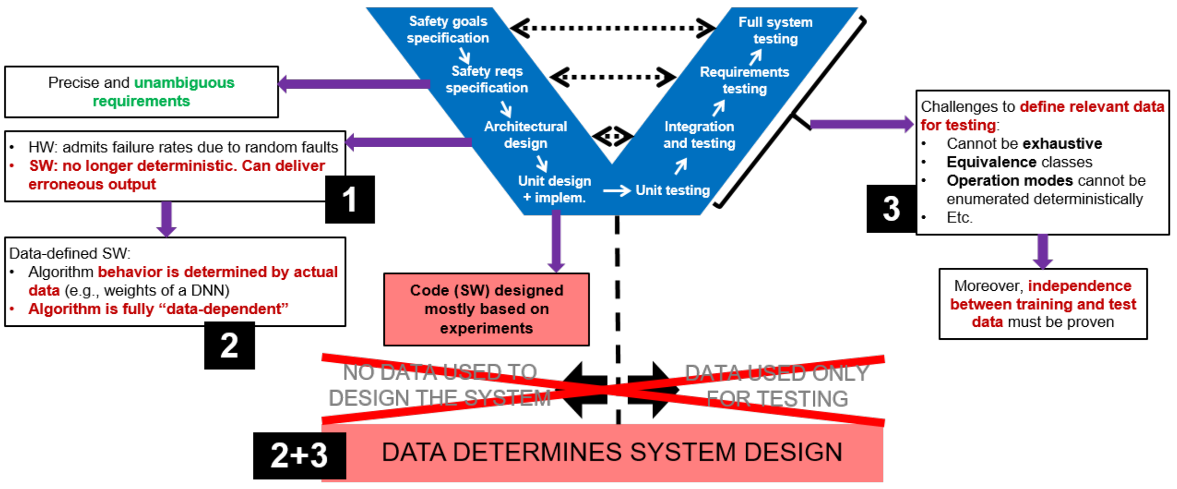
Safety goals specification
Safety reqs specification
Architectural design
Unit design + implem.
Unit testing
Integration and testing
Requirements testing
Full system testing
Precise and unambiguous requirements
HW: admits failure rates due to random faults
SW: no longer deterministic. Can deliver erroneous output
1
Data-defined SW:
Algorithm behavior is determined by actual data (e.g., weights of a DNN)
Algorithm is fully "data-dependent"
2
Code (SW) designed mostly based on experiments
Challenges to define relevant data for testing:
Cannot be exhaustive
Equivalence classes
Operation modes cannot be enumerated deterministically
Etc.
3
Moreover, independence between training and test data must be proven
~~NO DATA USED TO DESIGN THE SYSTEM~~
~~DATA USED ONLY FOR TESTING~~
2+3
DATA DETERMINES SYSTEM DESIGN

Fig. 1: Schematic of the challenges posed by DL solutions with respect to the conventional FUSA development process.

### *A. Specification and implementation of DL algorithms enabling explainability and traceability*

DL algorithms provide results with some degree of accuracy and confidence levels, as opposed to other types of control software used in FUSA-related systems, which provides deterministic and correct-by-construction outcomes (see 1 in Figure 1). Hence, software architectural design and verification methods do not longer work for systems building on DL algorithms in general since, even if DL software behaves as expected, outcomes can be inconclusive or simply erroneous. Moreover, non-DL software is developed in the form of algorithms were traceability is feasible and the different parts of the software at different abstraction levels (e.g., instruction, basic block, function) produce explainable results. Instead, DL software comes along with parameters generated from data through training, hence lacking explainability. Finally, since DL software is the result of a data-dependent training process, it behaves to a large extent as a black box, ultimately challenging traceability.

Identifying the challenges toward the certification of DL-based software is a very recent research topic. Along this line, some efforts have been devoted to reach a consensus on the definitions of those challenges in a systematic and consistent way, assessing the adherence of industrial and practically-implemented software, and providing guidelines toward resolving some of the identified issues according to domain-specific safety standards, in particular ISO 26262 [12], [13]. However, these efforts do not leverage explainability and end-to-end traceability of safety requirements as part of the implementation and specification of DL algorithms. Notably, these gaps are hindering the safe adoption of DL-based solution in CAIS. The work by Pullum et al. [14] provides some preliminary considerations and guidance on Validation and Verification (V&V) of neural networks. Also, Tabani et al. did an attempt to assess the adherence of an autonomous driving (AD) system against ISO 26262 requirements [15]. However, despite these preliminary attempts, the gap between DL and FUSA is still far from being filled.

### *B. Confidence, security, and robustness of DL algorithms*

Robustness of a DL-based system implies that the system needs to cope with noisy and uncertain input data, while also being able to express its confidence on its output even with unforeseen input data (see 2 and 3 in Figure 1). Robustness and security also require DL-based software to be robust against adversarial tampering with inputs [16], from random perturbations to universal adversarial patches to change image understanding altogether [17]. Current work in this field focuses on detecting targeted or non-targeted attacks on image input to various forms of Convolutional Neural Networks (CNN) in comparison to random noise added to an input [18]. Ensemble-techniques and uncertainty measures are promising starting points, but they also lack some precision and interpretability, in addition to inherent safety concerns [19]. Understandability and self-monitoring are elaborated upon in [20] where self-explaining models are developed in stages, progressively generalizing linear classifiers to complex yet architecturally explicit models.

### *C. Safety certification of DL software*

As of today, the state-of-the-art of AI solutions appears not to be reconcilable with safety and certification, as DL-based software development involves a major paradigm shift with respect to traditional system development and safety certification. Some recent work analyses the safety implications of AI, identifying some limitations and withholders that should be addressed in terms of method [12], [21], safety requirement specification, verification, validation, and testing [22], and overall safety lifecycle [23]. In the automotive domain, an earlier work [13] analysed the apparent incompatibility of ISO 26262 with DL and presented a set of initial recommendations on how to possibly improve such compatibility gap. The recently released ISO/PAS 21448 [9], a Publicly Available Specification (PAS) rather than a complete standard, acknowledges the incompatibility of DL and current standards (in the referenced domain), and provides general but insufficient guidance. Other related standards have also been very recently published (ANSI/UL 4600, ISO/TR 4804) or are currently under development (ISO/IEC TR 5469, ISO/AWI PAS 8800, ISO/AWI TS 5083, ISO/IEC AWI TS 6254).

### *D. Addressing performance and platform-level concerns for safely deploying DL software in CAIS*

The computational requirement of DL-based CAIS solutions can only be met with complex, high-performance, heterogeneous platforms. The use of advanced high-performance Commercial Off-The-Shelf (COTS) platforms is known to hamper the analyzability of a system [24], and to cause significant performance issues due to contention accessing hardware shared resources such as memories and caches, for which some limited solutions have been

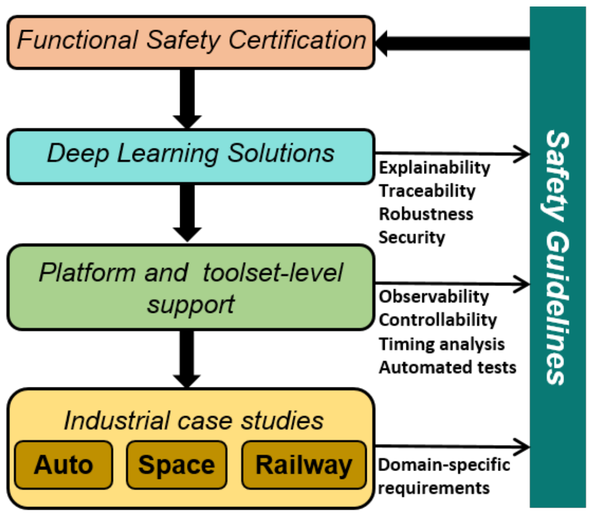

Fig. 2: SAFEXPLAIN methodology including its 4 pillars.

proposed recently [25], [26]. However, to our knowledge, no COTS hardware exists that allows excluding *all* sources of interference [27]. Some solutions have been proposed in different domains to deal with timing interference at software-level [28] but none of them has proven to be the one-fits-all solution, even for a single domain. The additional explainable DL structure, as an expected SAFEXPLAIN outcome, it is likely to require even more computing resources, especially for time-critical applications.

## IV. The SAFEXPLAIN Approach

In the following we illustrate the approach that SAFEXPLAIN will follow to tackle the macro-challenges identified in the previous section.

### *A. Specification and implementation of DL algorithms enabling explainability and traceability*

SAFEXPLAIN will devise a data-driven software specification approach for DL software, to develop advanced software where traditional software specification techniques fall short. In particular, such specification will consider how training data determines the behaviour of the algorithm, what steps are needed to process such data (specifically which DNN architecture is chosen), and how those steps allow justifying that DL components perform their functionality. Ultimately, such specification will support the end-to-end tracing of safety requirements.

### *B. Confidence, security, and robustness of DL algorithms*

SAFEXPLAIN will pursue the definition of a new generation of techniques for redesigning and adapting existing DL components and architectures with a view to improve explainability, traceability, robustness, and understandability. SAFEXPLAIN will also investigate on tools, models, model integration, and self-monitoring, for the engineering of CAIS. To achieve FUSA compatibility (including AI-related safety standards) for AI-based software, it is necessary to determine how the AI features of the systems shall be managed and, more concretely, how those features are handled within engineering and assurance tools, represented in models, and incorporated in supervisors and self-monitoring mechanisms, to ultimately make the qualification of those techniques possible.

### *C. Safety certification of DL software*

SAFEXPLAIN will take a significant step towards the reconciliation of FUSA and DL requirements, which remains as an open challenge. It will tackle the conflicting needs of integrating DL-based solutions for advanced functionality and the restrictive safety certification implications by following a FUSA-centric approach. While the prevailing literature in the topic passively attempts to map current AI research paradigms to standards, SAFEXPLAIN will proactively take selected safety standards as the basis to set the foundations on which to build its technologies, considering an end-to-end software development process (from design to validation, see Figure 1). SAFEXPLAIN will pave the way towards future certifiability of DL-solutions following an incremental FUSA strategy based on safety patterns for increasingly complex DL usage levels. The viability of the approach will be assessed and demonstrated through its integration in an industrial toolset prototype, hence providing evidence on the fact that SAFEXPLAIN solutions can reach TRLs above 5 in the future.

### *D. Addressing performance and platform-level concerns for safely deploying DL software in CAIS*

SAFEXPLAIN focuses on tailoring DL software and deploying it conveniently on COTS hardware while preserving the functional and non-functional features of DL software, and achieving the performance needed. Solutions will focus on adapting and optimizing DL software to deeply exploit hardware performance, on exploiting observability and controllability knobs of COTS hardware to increase predictability, and on the use of appropriate statistical and probabilistic analyses to attach guarantees to software timing predictions [29].

## V. Methodology

SAFEXPLAIN's methodology builds upon four pillars. Two main research pillars, where explainability and traceability play a central role for the certification of DL-based systems:

- Pillar 1: DL-aware Functional Safety Certification.
- Pillar 2: FUSA-aware Deep Learning Solutions.

As well as two supporting pillars linking pillars 1 and 2 with real tools, systems and applications:

- Pillar 3: Platform and toolset-level support.
- Pillar 4: Industrial case studies.

### *A. Overall methodology*

The development process of CAIS starts with the safety analysis, which we sketch next, since it tailors the methodology of SAFEXPLAIN.

Safety analysis identifies and assesses the system hazards and also defines mitigation strategies. FUSA standards define a safety life-cycle where each product item is assessed and managed from the functional safety standpoint: from analysis, specification, development, to operation and decommission. In the software development phase, standards refine the safety life-cycle in typical V-models (see blue central "V" in Figure 1). In these models, design phases, on the left side of the V, are matched to testing phases, on the right side, with implementation at the bottom.

The higher the criticality level – i.e., safety integrity level (SIL), or automotive SIL (ASIL) – of the software functionality, the more stringent the requirements posed by standards on the activities on each step of the V-model are. This reduces the risk of malfunctioning to the desired (negligible) level, i.e., probability of dangerous failure in the range of $10^{-9}$ hours of operation for SIL 4 safety functions.

The FUSA requirements placed on software emanate directly from the functional and technical safety requirements. Traceability is of paramount importance for high integrity functions where all design, implementation decisions, and tests shall be mapped to at least one requirement. Therefore, a well-formed specification must be made available for the software product (and parts thereof) as a baseline for assessing the correctness of the architectural and detailed design. The specification is expected to provide a detailed description of both static and dynamic aspects of execution, and to come up with trustworthy bounds on computational resource usage (timing, memory, communication). Under IEC 61508, certain properties are desirable for evaluating the systematic safety integrity of the software during safety requirement specification: completeness and correctness, freedom from intrinsic specification faults (including freedom from ambiguity), understandability and traceability of safety requirements and the capability of providing a basis for verification and validation (V&V) and associated testing.

In the implementation phase, the software is required to be correct with respect to its safety requirements, simple, readable, explainable, predictable and testable, which is suggested to be accomplished by following a restrictive set of coding rules and practice (e.g. no use of dynamic software features, as pointers, avoid recursion, enforce strong typing). Then integration gathers the different software parts and ensures they behave together as expected.

Verification (the system is right w.r.t. requirements) and validation (the system does the right thing w.r.t. its intended use) activities take a prominent role in the safety life-cycle. Hence, the importance of architecting and implementing software so that it is explainable, traceable, robust, secure and reliable.

While DL approaches offer a promising path to handle the increasingly complex functionalities handled by software in CAIS, several challenges are still unresolved before DL can be effectively adopted in safety-critical CAIS. The overarching approach in SAFEXPLAIN consists in tackling these concerns combining inter-disciplinary expertise, including AI, CAIS certification, and high-performance mixed-criticality hardware platforms with appropriate toolsets, studying and scrutinizing state-of-the-art case studies. In this line, SAFEXPLAIN builds on top of four main pillars: Functional Safety Certification, Deep Learning methods, Platform and Toolset-Level Support, and Case Studies. The main constituents of these pillars are represented in Figure 2.

## *B. Overall Methodology across Pillars*

FUSA will constitute the backbone of SAFEXPLAIN methodology: the project will introduce DL-software in safety-critical systems following an incremental methodology where DL is introduced in the system under analysis with different roles and relevance. Therefore, SAFEXPLAIN will start from exploring more conservative solutions, where the DL-software is not regarded as a safety component and can be integrated in critical systems based on the current versions of safety standards, up to the implementation of DL-based software for high criticality safety functions, where an update of current standards is foreseen.

Focusing on these scenarios, SAFEXPLAIN will generate the safety patterns based on a continuous analysis of safety standards, inputs from industrial partners and AI experts, and conversations with certification experts. Each pattern will shape the set of techniques, restrictions, requirements and rules to be adopted in SAFEXPLAIN.

For each identified safety pattern, SAFEXPLAIN will carry out the main loop shown in Figure 2, consisting in going through the following phases:

**FUSA techniques phase**: the specific safety pattern produces a set of high-level requirements and recommendations from the FUSA perspective. These requirements are developed into a set of specific requirements for the different phases in the development cycle of DL applications and libraries: specification and design, implementation, and V&V. With respect to the latter, SAFEXPLAIN will devise new testing methodologies to support the V&V activities for both functional and non-functional aspects of execution.

**DL algorithms and implementation**: the set of requirements emanating from the FUSA perspective are considered in the definition of novel or amended solutions in all aspects of development, still in support of the specific safety pattern. This phase will impose new requirements on DL software, to support explainability, traceability, robustness, security and fault-tolerance. Simpler patterns (e.g., not touching the safety function) will require fewer modifications to current practice: the constraints imposed under different SIL vary. For instance, in current standards, some features (e.g. use of pointers) are not recommended for high integrity level functions (e.g., ASIL C/D). Following the same philosophy, depending on the safety patterns – and hence the safety requirements on DL – different changes might be needed on the DL software to reach compliance. Hence, in this part of the work we will adapt DL software according to the specifications in the safety pattern and will evaluate the impact of those changes on performance.

**Platform and toolset level support**. Our approach includes both, functional and non-functional aspects.

Non-functional aspects (platform predictability): the set of FUSA requirements, in combination with the mixed-criticality requirements, will require the enforcement of different platform configurations for improved predictability and analyzability. Solutions must be provided to support the analysis techniques devised for the specific safety pattern. As a common preliminary step, the analysis and classification of the interference channels must be conducted on the selected platforms [30], [31]. At platform level, the impact that other applications (likely with different degrees of criticality) have on the application under analysis have to be controlled. Embracing a statistical view on residual software failure [29], for patterns in which DL software carries some safety requirements, may make a lower failure rate be required.

Functional aspects (industrial toolset for validation): DL-based solutions will be integrated into the industrial toolset, which will allow automating tests and analysing resulting data, such as, for instance, false positives and false negatives in object detection. Due to the novelty of these systems in safety-related industry, the industrial toolset is still at a prototype stage. Hence, DL-based solutions will be evaluated in a case-by-case direct integration and test campaign against each case study, as well as in the scope of the toolset, which will be in turn used with the case studies to find out the best integration approach that eases test automation and resulting data analysis.

**Case studies**: the results of the iteration of the SAFEXPLAIN methodology will be assessed on the industrial case studies for automotive, railway and space domains, providing valuable feedback on its effectiveness and industrial viability. The DL-based functionalities of the case studies focus on camera-based object detection in different scenarios and with different time and accuracy requirements since navigation in roads/streets, automatic train stopping and signal detection, and outer space navigation do pose highly diverse problems, despite their nominal similarity. For instance, object detection for trains must be highly specialized to recognize signals and unexpected objects in the railway. Instead, object detection in the outer space must be highly precise for docking and manage very dark environments. Finally, object detection for cars must consider the broadest range of scenarios, hence trading off precision across all of them. The case studies will be adapted as needed to match the considered FUSA scenarios, and to capture the peculiarities of the considered safety patterns identified. This allows assessing SAFEXPLAIN benefits under each pattern.

**FUSA guidelines**. After the iterations for all safety patterns are performed, the results of combining FUSA and DL requirements will be synthesized into a set of FUSA guidelines, and the technical solutions (or a subset thereof) will undergo a further external assessment. In particular:

*Safety guidelines*. The FUSA-related SAFEXPLAIN outcomes obtained throughout the project will be consolidated as safety guidelines. To this end, SAFEXPLAIN will follow a review process for both technical and concept assessments, first by internal certification experts from the project partner EXIDA, which will participate in the refinement of the certification approach, and finally by external certification experts with respect to selected industrial safety standards (e.g. IEC 61508, ISO 26262, ECSS and EN5012x) in the automotive, space and railway domains.

*Technical assessment and reviews*. SAFEXPLAIN selected technical contributions will undergo a review by Certification Experts (CE) in the automotive (ISO 26262, SOTIF), the space (ECSS) and railway (EN 5012x) domains. These reviews aim to assess the technical suitability of a subset of the safety patterns and a selection of FUSA techniques from the defined life-cycle activities.

**Concept assessment**. A selection of the SAFEXPLAIN FUSA-aware solutions and their integration on top of complex heterogeneous platforms in (mixed-) criticality systems will be then assessed with respect to safety certification by means of a safety concept based on a representative case study. This concept will include an analysis of security threats (e.g. adversarial attacks) and their impact on the DL algorithms and system safety. As a result, a concept review with the external certification experts and a consolidated set of safety guidelines will be produced with respect to selected industrial safety standards (e.g. IEC 61508, ISO 26262, SOTIF).

## VI. Conclusions

DL software stochastic and data-dependent nature is at odds with FUSA development processes, hence challenging the adoption of DL solutions in CAIS, as needed for autonomous operation. Therefore, new paradigms and practical approaches are needed to overcome this gap. SAFEXPLAIN will tackle this challenge by building DL solutions providing explainability and traceability as the backbone to enable FUSA-compliant development processes. Moreover, SAFEXPLAIN will overcome the limitations imposed by current DL-unaware FUSA standards providing DL-aware FUSA guidelines that allow using DL solutions in a wide variety of safety patterns with varying requirements preserving properties such as high accuracy and confidence of the best DL solutions, as well as sufficiently high performance to enable their use in performance-hungry applications such as, for instance, camera-based object detection for driving and navigation in domains such as automotive, space and railway.

## Acknowledgements

The research leading to these results has received funding from the Horizon Europe Programme under the SAFEXPLAIN Project (www.safexplain.eu), grant agreement num. 101069595. BSC authors have also been supported by the Spanish Ministry of Science and Innovation under grant PID2019-107255GBC21/AEI/10.13039/501100011033.

## References

[1] J.F. McCarthy, "Sustainability of Self-Driving Mobility: An Analysis of Carbon Emissions Between Autonomous Vehicles and Conventional Modes of Transportation. Master's thesis, Harvard Extension School. ," 2017.

[2] Y. LeCun et al., "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[3] "Apollo, an open autonomous driving platform." http://apollo.auto/, 2018.

[4] NVIDIA, "NVIDIA Drive program," https://developer.nvidia.com/drive, 2018.

[5] International Electrotechnical Commission, *IEC 61508. Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related Systems*, 1998.

[6] International Standards Organization, *ISO/DIS 26262. Road Vehicles — Functional Safety*, 2009.

[7] European Committee for Electrotechnical Standardization (CENELEC), *EN 50126 - Railway Applications - The Specification and Demonstration of Reliability, Availability, Maintainability and Safety (RAMS)*, 2017.

[8] ——, *EN 50128 - Railway applications - Communication, signalling and processing systems - Software for railway control and protection systems*, 2020.

[9] International Organization for Standardization (ISO), *ISO/PAS 21448 Road vehicles - Safety of the intended functionality*, 2019.

[10] Underwriters Laboratories (UL), *ANSI/UL 4600 Standard for Safety for the Evaluation of Autonomous Products*, 2020.

[11] SAE International, *J3016: Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems*, 2014.

[12] P. Koopman and M. Wagner, "Autonomous vehicle safety: An interdisciplinary challenge," *IEEE Intelligent Transportation Systems Magazine*, vol. 9, no. 1, pp. 90–96, 2017.

[13] R. Salay et al., "An analysis of ISO 26262: Machine learning and safety in automotive software," *IEEE Intelligent Transportation Systems Magazine*, no. SAE Technical Paper 2018-01-1075, 2018.

[14] L. Pullum et al., *Guidance for the verification and validation of neural networks.* John Wiley and Sons, 2007.

[15] H. Tabani et al., "Assessing the adherence of an industrial autonomous driving framework to iso 26262 software guidelines," in *DAC*, 2019.

[16] C. Szegedy et al., "Intriguing properties of neural networks," 2013. [Online]. Available: https://arxiv.org/abs/1312.6199

[17] T. Brown et al., "Adversarial patch," 2017. [Online]. Available: https://arxiv.org/abs/1712.09665

[18] A. Fawzi et al., "Robustness of classifiers: From adversarial to random noise," in *NIPS*, 2016.

[19] M. Abbasi and C. Gagné, "Robustness to adversarial examples through an ensemble of specialists," 2017. [Online]. Available: https://arxiv.org/abs/1702.06856

[20] D. Alvarez-Melis and T. S. Jaakkola, "Towards robust interpretability with self-explaining neural networks," in *NIPS*, 2018.

[21] J. Hernández-Orallo et al., "Surveying safety-relevant AI characteristics," in *SafeAI Workshop*, 2019.

[22] P. Koopman and M. Wagner, "Challenges in autonomous vehicle testing and validation," *SAE International Journal of Transportation Safety*, vol. 4, no. 1, pp. 15–24, 2016.

[23] A. Pereira and C. Thomas, "Challenges of machine learning applied to safety-critical cyber-physical systems," *Machine Learning and Knowledge Extraction*, vol. 2, no. 4, pp. 579–602, 2020.

[24] J. Abella et al., "WCET analysis methods: Pitfalls and challenges on their trustworthiness," in *SIES*, 2015.

[25] S. Girbal and J. Le Rhun, "BB-RTE: a Budget-Based RunTime Engine for Mixed and Safety Critical Systems," in *ERTS*, 2018. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02278298

[26] A. Kritikakou et al., "Distributed run-time WCET controller for concurrent critical tasks in mixed-critical systems," in *RTNS*, 2014.

[27] J. Pérez-Cerrolaza et al., "Multi-core devices for safety-critical systems: A survey," *ACM Comput. Surv.*, vol. 53, no. 4, pp. 79:1–79:38, 2021.

[28] S. Girbal et al., "Deterministic platform software for hard real-time systems using multi-core COTS," in *DASC*, 2015.

[29] F. J. Cazorla et al., "Probabilistic worst-case timing analysis: Taxonomy and comprehensive survey," *ACM Comput. Surv.*, vol. 52, no. 1, pp. 14:1–14:35, 2019.

[30] I. Agirre et al., "On the tailoring of CAST-32A certification guidance to real COTS multicore architectures," in *SIES*, 2017.

[31] F. J. Cazorla et al., "Dissecting robust resource partitioning, robust time partitioning, and robust partitioning in CAST-32A," *SAE Technical Paper 2021-01-5101.*, 2021.