From Principles to Practice: An End-to-End Approach for Trustworthy ML in Critical Systems

Afef Awadid, Lucas Mattioli, Karla Quintero
IRT SystemX, France
email: {afef.awadid, lucas.mattioli, karla.quintero}@irt-systemx.fr

Juliette Mattioli
Thales, France
email: juliette.mattioli@thalesgroup.com

***Abstract*—This work presents one of the products of the Confiance.ai research program which addresses an end-to-end method for engineering trustworthy ML-based systems [1]. The proposed methodology revisits software and systems engineering as it encompasses all development phases of the system while integrating the specificities related to the development of ML-based components within the system. The method leverages vastly researched and deployed standard procedures from design to validation and maintenance in order to provide rigor, structure, and traceability when developing ML-models.**

Keywords- trustworthy AI; trustworthiness attributes; trustworthiness risk analysis; trustworthiness assessment; safety-critical ML-based systems; end-to-end engineering methodology; trustworthy AI engineering.

I. Introduction

The term "AI" (Artificial Intelligence) was first used in a workshop held at Dartmouth College in 1956. It was introduced as a branch of computer science that tries to mimic human thinking by using symbols and knowledge bases that are also symbol-based. Any technology, even AI, is developed to provide a service fulfilling some needs. The AI discipline aims to embed cognitive capacities such as perception, learning, reasoning, planning, decision and dialogue, to an artificial system. In February 2025, the European Commission defines an AI system (see Figure 3) as the following: *a machine-based system that is designed to operate with varying levels of autonomy that may exhibit adaptiveness after deployment and, for explicit or implicit objectives, infers from the input it receives how to generate outputs, such as predictions, content, recommendations or decisions that can influence physical or virtual environments*.

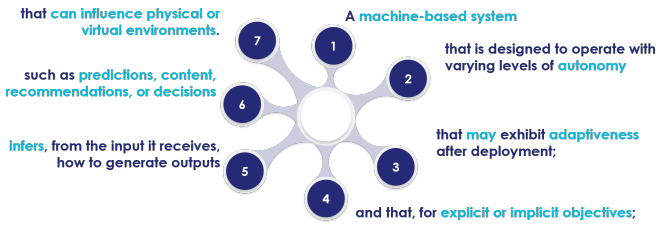

Figure 1. The European Commission AI-system definition comprises seven main elements (Feb. 2025)

In our context, an AI-based system is defined as a system that incorporates AI components. AI-based critical systems, which can have severe consequences in case of failure, are considered to be "high risk" under the European regulation known as the "EU AI Act" [2]. These systems can, for example, represent safety components of regulated products which are required to undergo a third-party conformity assessment. Examples of such systems can be found in the fields of transportation, healthcare, defense, and security in general. The deployment of such systems is contingent upon their demonstrated capacity to deliver the anticipated service in a secure manner, while meeting user expectations with regard to quality and continuity of service. Furthermore, users might consider negative any surprising or unexpected actions from the system.

In order to characterize such systems with a view to quality assurance, [3] proposed considering several dimensions, including the artifact type dimension, the process dimension and the trustworthiness characteristics, which are relevant to software product or system quality. Furthermore, the focus of the series of standards SQuaRE (Systems and Software Quality Requirements and Evaluation) is on software quality [0]. In addition, the specific nature of AI is addressed in order to provide a quality model for AI systems. Consequently, the design of critical AI systems requires the demonstration of their trustworthiness, as asserted by [4].

Trustworthy AI is based on these three components [5]: it must comply with all applicable laws, adhere to ethical principles, and be robust. This shift is driving the new discipline of AI engineering [6] to support the industrial design of such systems. Therefore, the development of AI-enabled systems is heavily dependent on the application of specific traditional software and system engineering practices. For example, engineering teams must conceptualize AI systems that can handle the inherent uncertainty of their components, data, models, and outputs — particularly when implementing data-driven AI. The user experience with AI systems is dynamic [7]. Interfaces must clearly show what the system is doing, how outputs are generated (dataset quality), and when the system is not behaving as expected (monitoring throughout the lifecycle). Therefore, engineering teams must account for the different rhythms of change, including changes in data, models, systems, and business processes.

This discipline aims to ensure that critical AI-based systems in safety-, mission- and business-related domains are valid, explainable, resilient, safe, secure, and compliant with regulations, standards, and responsible practices (ethics, sustainability, etc.). When dealing with critical systems, additional

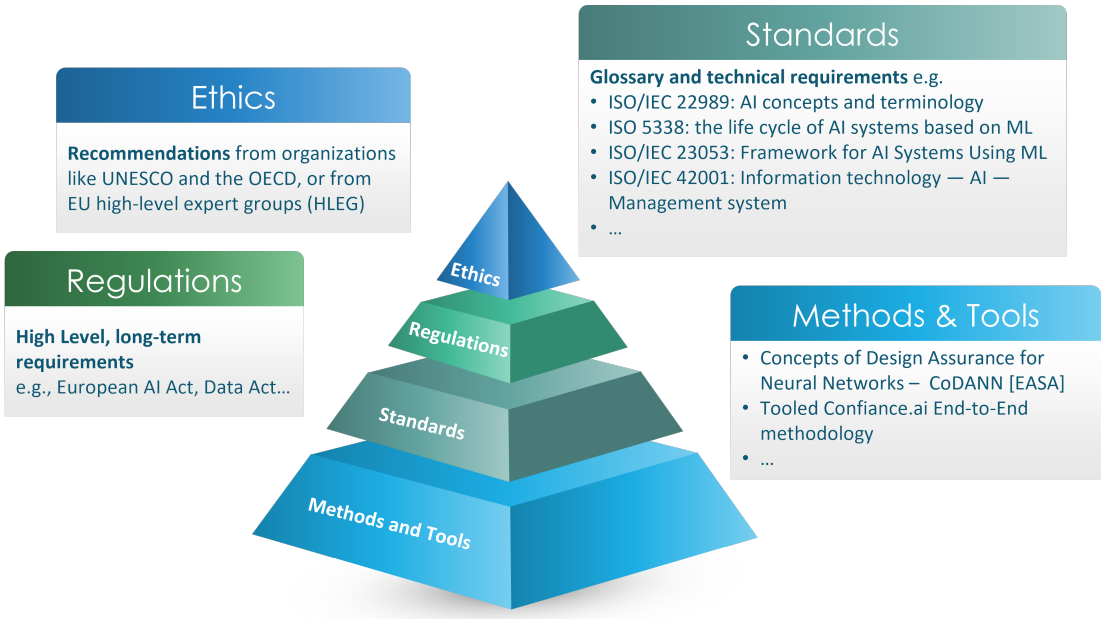

Figure 2. From ethics to the end-to-end methodology through regulation and standards

constraints must be considered. In the context of system design, processes must be optimized, justified, replicated where possible, and improved. However, it is also essential to ensure that the system meets the appropriate level of trustworthiness [8] [9]. This includes robustness (the ability of a system to withstand errors during execution and cope with erroneous input), cybersecurity, and dependability (including reliability, availability, maintainability, and safety properties), among others.

Thus, in the following, we will first remind the current context of AI regulation and standardization of the definition of "trustworthiness" as "*the ability to meet stakeholders' expectations in a verifiable way*". To determine whether a given risk is as such, it should be first identified. Then it should be analyzed with respect to the "intended purpose" which is defined as "*the use for which an AI system is intended by the provider". It also includes "the specific context and conditions of use*".

It is imperative that a well-established engineering discipline oversees AI capabilities. "**AI Engineering**" is an emerging discipline that focuses on applying AI in real-world contexts. AI engineering involves applying engineering principles and methodologies to create scalable, efficient, trustworthy and responsible AI-based solutions. It merges aspects of data engineering, knowledge engineering, algorithm engineering, software engineering, system engineering, cyber-security, safety and ethical engineering and also cognitive engineering including human factors to accelerate the development and the deployment of AI-based capabilities. It also speeds up the maturation of individual tools. This is particularly evident in high-stakes scenarios such as responding to national security threats and military operations. Therefore, to maximize the potential of AI in such situations, we must address the unique challenges that AI systems encounter. While the capability to develop AI systems has increased due to greater computing power and more extensive datasets, these systems often only function in controlled environments and are difficult to replicate, verify and validate in real-world scenarios. AI Engineering aims to provide a framework and tools to proactively design AI systems to function in environments characterized by high degrees of complexity, ambiguity, and dynamics.

Then, we present an end-to-end methodology to support "trustworthy AI engineering", which encompasses the entire lifecycle of AI-based systems, from operational design domain (ODD) specification to maintenance [1]. This holistic methodology covers the design, development and deployment of AI systems in critical environments, including data engineering, algorithm design and development, deployment and monitoring. By integrating the principles outlined in international and national initiatives with our advanced internal engineering practices, this lifecycle ensures that AI systems perform their intended functions with the desired level of performance. It also makes AI-powered solutions transparent, responsible, and ethical. This systematic approach involves organizing multidisciplinary and fragmented approaches to trusted AI and applying a continuous workflow. Measures to improve AI trustworthiness must be implemented at every stage, including data sanitization, robust algorithms, anomaly monitoring, and risk auditing.

II. Regulation and Standardization

To ensure safety, reliability, availability, and maintainability, AI systems must perform, and continue to perform, as intended under sufficient conditions. Hazard analysis and risk assessment must be tailored to the unique characteristics of AI systems. This includes identifying potential critical errors in

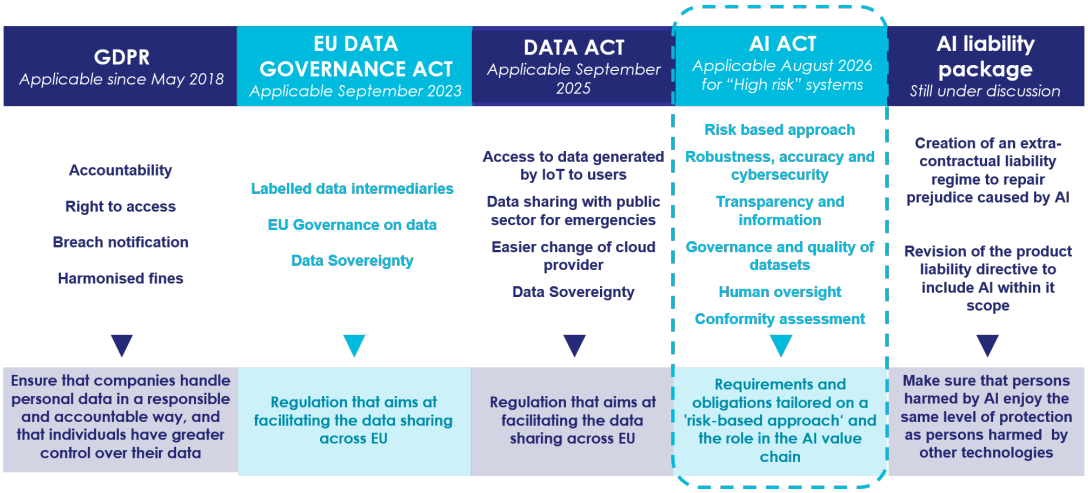

Figure 3. European Data and AI Regulation

the training data or knowledge representation and assessing the AI model's ability to generalize to unseen operational data. Performance requirements for AI algorithms are often driven by safety objectives, which limit the worst credible approximation error to an acceptable threshold.

However, trustworthiness is closely linked to accountability, which can be considered a measure of trust or an alternative to it. In [10], dependability is used to represent a system's overall quality based on four sub-attributes: security, safety, reliability, and maintainability. Subsequently, security and dependability became key attributes of trust in computer-based systems [11].

Another requirement relates to the quality of datasets used to train, validate and test models for high-risk AI systems. This considers a non-exhaustive list of issues, such as data collection processes, data engineering activities (*e.g.*, annotation, labeling, cleaning, enrichment and aggregation), data quality assessment and identification of possible data gaps or shortcomings and how these can be addressed. Last but not least is the mitigation of possible biases likely to affect the health and safety of individuals or lead to discrimination.

In 2019, the U.S. National Artificial Intelligence Research and Development Strategic Plan [12] stressed the importance of standard metrics for quantifying AI technologies. "Standard metrics are required to define quantifiable measures in order to characterize AI technologies". As a matter of fact, [13] have recently stated that "a great deal of effort is required to determine which suitable measurements should be utilized to evaluate system performance across characteristics for responsible AI and across profiles for specific applications/contexts". Governments are responding with regulations typically associated with human rights. In 2024, the European Union adopted the AI Act (see Figure 3). These regulations set out long-term, high-level requirements, sometimes based on recommendations from organizations such as UNESCO [14] and the OECD [15] [16], or from High-Level Expert Groups (HLEG) [5].

These high-level requirements need to be operationalized for companies and developers. As shown in Figure 3, standards and regulatory frameworks define more detailed requirements, but they focus on what to do rather than how to do it. This leaves the choice of tooled, end-to-end methodology for developing AIs that fulfill these requirements to companies and developers.

The Assessment List for Trustworthy AI considers 7 pillars of trustworthiness: 1) human agency and autonomy, 2) technical robustness and safety, 3) privacy and data governance, 4) transparency, 5) diversity, non discrimination and fairness, 6) societal and environmental well-being, 7) accountability. This List is one of the basis of the AI Act [2] which requires companies to take measures to ensure that their products developed or deployed in the European Union are safe and comply with ethical principles.

In the aeronautic domain, EASA [17] proposes a model of trustworthiness based on the characterization of the machine learning (ML) application (high-level function/task, concept of operations, functional analysis, classification of the ML application), safety assessment, information security management, and ethics-based assessment (which includes the 7 pillars of the ALTAI [18]). The Fraunhofer [19] offered an analysis of the standard [20] on management system for AI, stating compliance to the standard can contribute to ensuring AI trustworthiness since it encompasses the pillars of the ALTAI, provided that a third-party verification has been performed and along with an adapted quality management system.

In the same period, the characteristics of trustworthy AI system specified by the NIST include: "*valid and reliable, safe, secure and resilient, accountable and transparent, explainable and interpretable, privacy-enhanced and fair with harmful bias managed*". Then the NIST produced an analysis of the components of trust [21] and highlighted several top level aspects for the design of a trustworthiness model, that should encompass the user experience, the perceived technical trustworthiness, the pertinence of each trustworthiness characteristic in the user's specific context of use...

Standards provide a framework for legislation and rules by recording the current state of the art and recommended

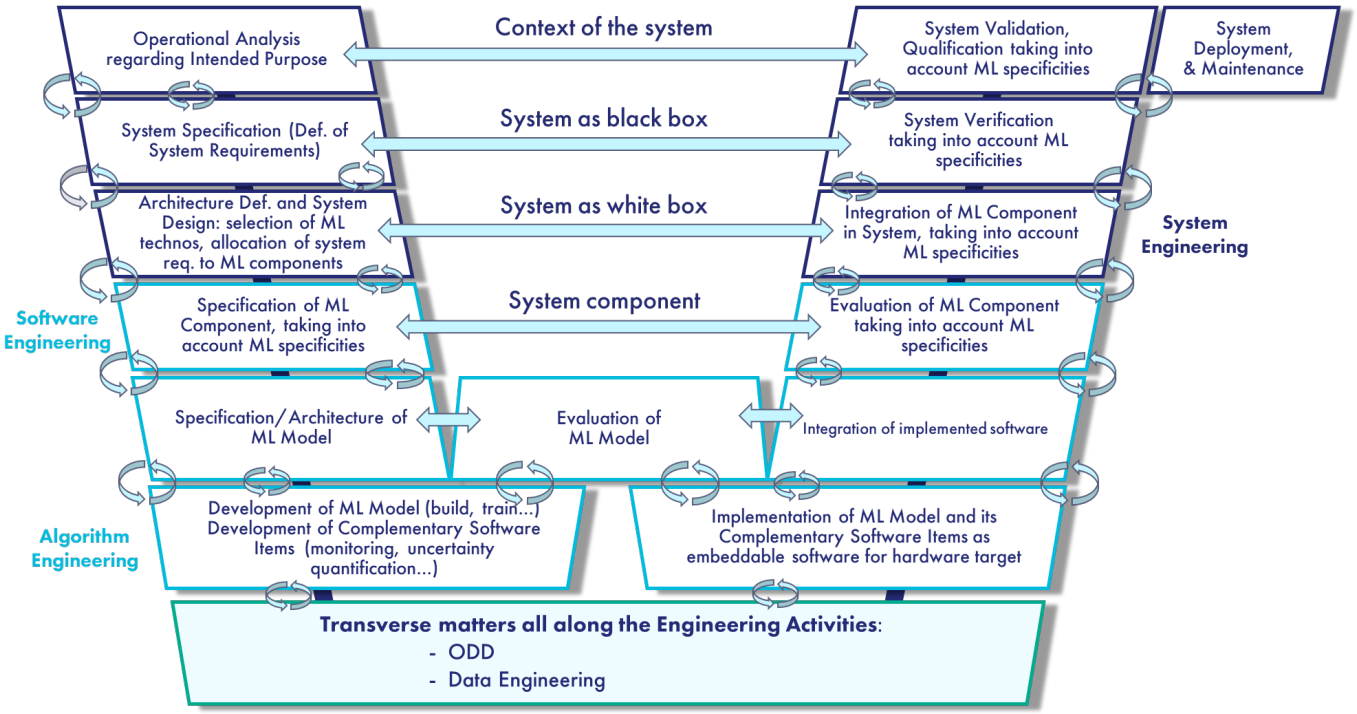

Figure 4. High-level view of the end-to-end methodology (source: https://bok.Confiance.ai).

practices, and offering a foundation for showing adherence and accreditation. Several organizations and initiatives, such as ISO/IEC, IEEE and NIST, are currently working on developing relevant AI standards. The standards developed by ISO/IEC [20] cover a wide range of AI aspects. These include terminology, performance metrics, data quality, ethics, and human-AI interaction. The ethical implications of AI technologies are the focus of the IEEE P7000 series of standards [22]. The NIST framework [21], meanwhile, provides guidance on managing risks, ensuring data quality, and promoting transparency and accountability in AI systems.

In 2019, ETSI set up an Industry Specification Group on Securing AI (ISG SAI) [23] to provide existing and potential mitigation against threats for AI-based systems. Robust security measures must protect AI systems from cyber-attacks, data breaches and unauthorised manipulation. These should include advanced threat detection and mitigation strategies and resilience mechanisms to operate securely in hostile environments. Cybersecurity should be embedded in the system and data pipelines. The lines between security and safety are not always clear when it comes to AI. Incorrect outputs can be caused by malicious actions or natural events.

Ethical engineering focuses on the need for fairness, transparency, and accountability in AI. This involves ensuring that algorithms are unbiased, produce explainable results, and adhere to societal and legal values. This engineering requires ongoing review by engineers, ethicists and domain experts.

However, it is crucial to recognize that the transfer of AI technology, particularly ML, must adhere to specific standards and processes to successfully transform research outcomes into industrial products fit for purpose that meet customer needs. For example, since data collection and analysis are crucial for developing any ML-based system, prioritizing data quality is essential. This requires adherence to compliance regulations, such as those relating to data privacy. Concurrently, operational requirements encompassing maintenance must be addressed. It is therefore evident that developing and implementing AI/ML systems involves both technical and business aspects, from problem conception to customer delivery. The development and operation of critical AI systems therefore requires an end-to-end, tool-based AI engineering methodology, which will be outlined subsequently.

III. The End-to-End Methodology

The version of the methodology presented herein has been produced as a result of the work within the French program Confiance.ai [1] [25] [26] [27] which was a pillar of the Grand Défi "AI for industry" initiative, which is pioneering methodologies for the development of trustworthy AI systems across sectors. Its associated roadmap is nourished by industrial needs and the evolution of the state-of-the-art [28]. Namely, several industrial projects and research initiatives have derived from Confiance.ai, generating the emergence of an ecosystem for the engineering of trustworthy AI for critical systems. In addition, the **European Trustworthy AI Association** (https://www.trustworthy-ai-association.eu/) is built on an open-source, community-driven approach, serving as a key enabler, giving stakeholders access to a dynamic ecosystem where they can learn from peers and co-develop tools. These tools are designed to ensure the adoption of

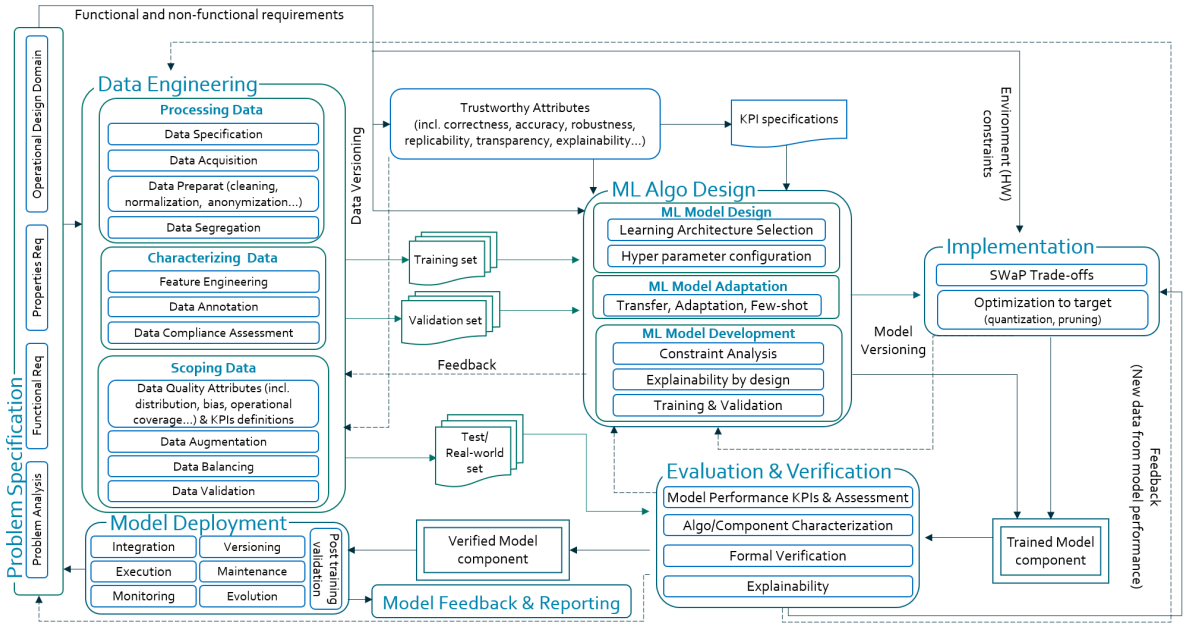

Figure 5. ML algorithm engineering pipeline [24]

scalable, secure and trustworthy AI based on this end-to-end methodology, which supports the engineering of trustworthy AI-based systems. The proposed methodology addresses following issues [29]:

- How can AI/ML models be designed to satisfy trustworthy attributes (explainability, robustness, accuracy, etc.)?
- How can these models allow a clear understanding of their behavior in the operational domain?
- How can AI/ML models be implemented and embedded on hardware, by making them fit to the target without discarding their trustworthy properties?
- Which data engineering methods should be applied to manage large volumes of data and account for the evolving operational domain?
- What kinds of verification, validation, and certification processes should be considered when dealing with AI/ML-based systems?

By addressing these challenges, the end-to-end methodology aims to answer the research question: How to ensure the reliability and trustworthiness of AI-based safety-critical systems? It is based on the premise that the development of ML-based critical systems should be structured with a trustworthiness imperative from the design phase, thus providing precise requirements for integration, verification, and validation, as well as for proper deployment and maintenance [30] [31]. It is a multi-domain collaboration that leverages concepts and procedures coming from different fields into the agnostic proposal of engineering trustworthy AI/ML-based critical systems. The result is the formalization, through a common language, of the structure and workflow for all actors involved in the process of designing trustworthy AI-based critical systems, *i.e.,* data engineers, systems engineers, safety engineers, software engineers, among many others.

A. ML Algorithm Engineering

The engineering ML-based systems is often portrayed as involving the creation of an ML model and its deployment. But in practice, the ML model is only a small part of the whole system. Much more is needed to ensure that an ML model is trustworthy and its behavior is predictable. This includes things like designing data pipelines, monitoring and logging, and so on. We defined the ML algorithm engineering pipeline to capture these aspects of AI engineering (see Figure 5). This pipeline differentiates between three types of development: requirements-driven, outcome-driven and AI-driven [32]. The starting point is that data must be available for training. Data engineering provides the foundation for various data collection and qualification methods, which can then be divided into training, testing, and cross-validation sets.

The following steps are encapsulated as sub-tasks within the pipeline:

1) **Problem specification**: Inclusion of the operational design domain (ODD), which is the description of the specific operating condition(s) in which a safety-critical function or system is designed to operate properly, including but not limited to environmental conditions and other domain constraints [33]. These requirements and architecture are the result of subsystem design activities and are part of specification activities. These requirements describe the specific function that the ML items should implement. They also describe the safety, performance and other requirements that the machine learning items should achieve.

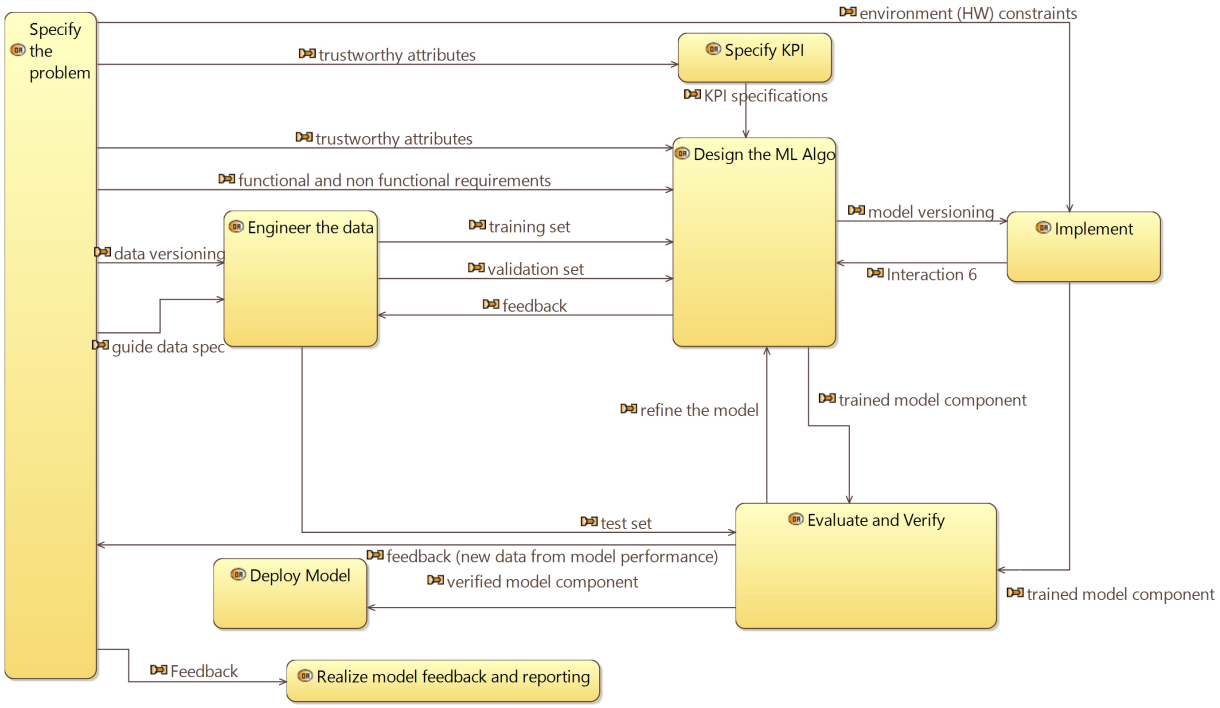

Figure 6. ML algorithm engineering process

2) **Data engineering**: Large amounts of data are needed to train a machine learning model so that it can learn to carry out its function. Before it can be used, data needs to be collected. It also needs to be prepared. The process of collecting data from a number of sources is known as data aggregation. The collected data needs to be substantial, accessible, comprehensible, reliable and usable. The process of transforming raw data into usable information is known as data preparation, or data preprocessing.
3) **ML Algorithm Design**: The ML algorithm can be trained using a feeding set, allowing it to learn appropriate parameters and features. The model will be refined once the training is complete, using the validation dataset. This process may involve modifying or discarding variables, as well as tweaking model-specific settings (hyperparameters) until an acceptable level of accuracy is achieved.
4) **Implementation**: For an ML component to be developed, the targeted hardware platform, the IDE (Integrated Development Environment) and the language for development must be decided on. There are a number of options to choose from. The majority of these would undoubtedly satisfy our requirements, as all of them provide the implementation of AI algorithms that have been discussed to date. However, it is sometimes necessary to take into account embedded constraints.
5) **Evaluation and verification**: Once we have found an acceptable set of hyperparameters and optimized the model's accuracy, we can test it. The testing process employs our test dataset and is intended to verify that our models are utilizing accurate features. We may decide to retrain the model based on the feedback we receive. This could lead to improvements in accuracy, adjustments to output settings, or the deployment of the model as required.
6) **Model Deployment**

The next step is to design or select an AI algorithm from an existing library (*e.g.*, Scikit Learn [34]) to create a model. The model is then trained iteratively so that the result closely aligns with the "correct answers" from the ground truth. The model is then ready to be deployed, like any other component, once validation has been successfully completed, provided that the specified criteria have been met.

As algorithm engineering workflow, ML pipelines consist of several steps to train a model (see Figure 6). Such pipelines are iterative as every step is repeated to continuously improve the accuracy of the model and achieve a successful algorithm. Pipelines are not one-way flows. They are cyclic in nature and enables iteration to improve the scores of the machine learning algorithms and make the model scalable.

The method addresses as a whole both the system engineering layer and the ML algorithm engineering layer. The system layer accounts for all underlying phases that should design and specify to further along verify and validate the overall system's objective and performance as carried out in classic systems engineering. The ML layer then covers all phases related to the ML component that inherit system requirements to then refined requirements specific to the ML-components to be developed. This process aims to ensure the compliance of

the AI/ML components with the overall system requirements and intended purpose.

B. Data Engineering

The quality of the data-set depends on the processes and technologies that ensure data values are consistent with the ODD. These values are then assessed for quality using various methods and key performance indicators (KPI) [35]. In certain instances, when a natural language processing application contains spelling errors, it is possible to ascertain the caliber and dimensions, such as accuracy [36]. However, it is more difficult to detect admissible but incorrect values. As discussed in [37], most ML research focuses on improving model performance rather than datasets. Typically, classical ML practices involve using existing datasets and enhancing the complexity of techniques to address performance challenges. Conversely, data-driven AI adopts a more comprehensive strategy, placing significant emphasis on the data itself [38] [39]. Instead of just looking for patterns and relationships in the features that are given, data-driven AI involves collecting, processing and analyzing lots of data to create models that are more accurate and robust. Furthermore, it is a real challenge in the present day to link datasets together. These should be linked to the ODD. This should be done at the operational level of the system definition.

As highlighted in [40], dataset quality may have a bigger impact on performance than model design. Poor data quality is a major risk in data-driven AI, as it can cause issues at every data engineering step, like collection, annotation and feature engineering, and can lead to problems being missed. To overcome these challenges, the Confiance.ai research program proposes a methodological process for assessing data trustworthiness.

Data trustworthiness evaluation indicates the degree to which data and data items satisfy expectations. An overview of the main metrics used for the data quality assessment is summarized in Figure 7. This evaluation can be carried out at various stages of the process, typically during data development (*e.g.*, raw data or dataset preparation), but also during IVVQ and deployment (*e.g.*, to detect data drift).

C. ML Algorithm Design

In this phase, a modeling technique is chosen and applied, and its parameters are set, then an ensemble model is developed and tested. The variant and structure type are both determined here, as is the algorithm. This process is referred to as "training", wherein data and outcomes are employed to optimize the configuration of the model. This process constitutes the "learning" aspect of ML.

Various ML techniques are available. These include multiple types of classification models. These models identify the category that the input belongs to. There are also regression models. These predict a continuous-valued attribute for supervised tasks. Then there are clustering models. These group similar items into sets for unsupervised tasks. Finally, there are reinforcement learning models. These provide an optimal set of actions.

A common question is "*Which ML architecture should I use?*". The following table [41] is provided by the DEEL project (https://www.irt-saintexupery.com/deel/), which summarizes the most common ML techniques and their main applications. Each ML technique relies on one or more hypothesis function spaces and one or more exploration algorithms (not listed in this document) to minimize a loss function on the training dataset.

Techniques	Applications
Linear models: Linear and logistic regressions, SVM	Classification, Regression
Neighborhood models: KNN, K-means, Kernel density	Classification, Regression, Clustering, Density estimation
Trees: decision trees, regression trees	Classification, Regression
Graphical models: Bayesian network, Conditional Random Fields	Classification, Density estimation
Combination of models: Random Forest, Adaboost, XGboost	Classification, Regression, Clustering, Density estimation
Connexionist and statistical models: Neural networks, Deep learning...	Classification, Regression

After choosing the model, among the various algorithms present, one needs to tune the hyper parameters of each model to achieve the desired performance.

- Select the right algorithm based on the learning objective and data requirements.
- ConFigure and tune hyperparameters for optimal performance and determine a method of iteration to attain the best hyperparameters.
- Identify the features that provide the best results.
- Determine whether model explainability or interpretability is required.
- Develop ensemble models for improved performance.
- Test different model versions for performance.
- Identify requirements for the overall lifecycle.

The resulting model can then be evaluated to determine whether it meets the business and operational requirements.

D. The ML-based system lifecycle

Developing ML-based systems can be visualized as a "W-shaped" life-cycle (see Figure 4). This W-shape can be split into two parts. For AI systems, "intended goal"/"intended purpose" and "intended domain of use" are very high-level requirements that have to be translated into "engineering terms". The engineered "intended domain of use" is called Operational Design Domain (ODD). The ODD is the operational conditions for which an AI system is specified, designed, verified, assessed, operated, and disposed. ML engineering life-cycle begins with defining AI/ML algorithm requirements refined from system specification. This ML specification step includes the characterization of the ODD.

	Approach / method	Machine Learning use	Data
Diversity	DPP [42]	Training and test sets	Image (object localization) & text (document summarization)
	R-DPP [43]	proposition of new metric	No data implemented
	D-MCL [44]	Training & test sets	Image (classification)
	PDS [45]	Training & test sets	Image (MNIST [46]), audio & 2-D synthetic data
Completeness	MADI [47] [48]	Not used for ML	Numerical & categorical data (from hospital for [48])
	POVM [49]	Test set (evaluation by using deep learning approaches (ICCnet & FidNet))	Tomography images
	MCAR, MAR, NMAR [50]	Training & test sets	Numerical data (diabetic data)
Representativeness	R-indicator [51] [52]	Not used for ML	Internet Data sources for [52]
	CI [53]	Test set	Tabular data
	Log Disparity [54]	Sampling, training and test sets	Clinical trials (classification)
Coverage	SelectiveNet [55] [56]	Training and test sets	Image (MNIST, Cifar [57] & ImageNet)
	Neuron Coverage [58]	Test set	Image (MNIST, ImageNet, Driving datasets) & numerical/categorical (Contagio, VirusTotal & Drebin datasets)
	DeepTest based on neuron coverage [59]	Test set	Image (real driving camera & synthetic)
	TensorFuzz [60]	Test set	Image (MNIST)
	TDA-AI2 [61]	Test set (applied on DRL)	3-D cloud data
Corner cases	[62], [63], [64],[65],[66],[67]	Anomaly detection	Images or videos
	[68]	Anomaly detection	Images (optical, radar & lidar)
	[69]	Anomaly detection	Numerical data (trajectories)
	[70],[59]	model evaluation	Images
	[58]	model evaluation	Images, PDFs, Android apps

Figure 7. A brief overview of the approaches and metrics used for data quality evaluation

This engineering activity is a critical step that changes the way AI researchers and engineers work. It involves a detailed description of all possible operating conditions, called the operating environment of the system, to enable data collection and knowledge representation. The reliability of the AI-based system depends on the correctness and completeness of this description, particularly for rare events or combinations of conditions that could be unsafe. The validity of a system is established by its intended use [71]. The ODD description is developed using a combination of top-down and bottom-up approaches. ODD aligns data and functional intent, *i.e.*, the data used for training and the resulting ML model(s) with their intended use, covering a wide range of conditions.

Data engineering is key. It involves the identification, collection, preprocessing and extraction of features from large datasets. These datasets are essential for designing and verifying ML models. This phase often involves advanced techniques. These techniques improve the representativeness, completeness, and relevance of the dataset (minimizing the simulation-to-reality gap). Rigorous quality controls, guided by Data Quality Requirements (DQRs), ensure data inputs are accurate and consistent. During model design, engineers select appropriate learning algorithms and improve model architectures through training and evaluation cycles. Optimization strategies balance computational efficiency and performance.

The second "V" of the "W-shaped" life-cycle includes the implementation engineering processes performed on the target platform (*e.g.*, specific hardware embedded in a ground or aerial vehicle). Validation and verification activities are driven by key trustworthiness properties, specified in low-level ML requirements. Validation activities ensure the correctness and completeness of ML requirements by verifying, analyzing, and tracing them back to higher-level requirements. Verification activities include extensively simulating, testing edge/corner robustness, scenario-based testing, analyzing the ML model explainability, and ODD coverage analysis [72]. The first level of verification ends with a selected AI model, which meets all its requirements in the development (learning) environment and serves as a design specification, ready for implementation into software and/or complex electronic hardware elements in the second level of verification. Figure 8 shows a high-level view of the verification phase of an automated feature based on ML and the interaction with the specification and validation phases.

MLOps, or Machine Learning Operations, and AI Engineering, while closely related, serve distinct roles within the ML lifecycle. MLOps focuses on the operationalization of machine learning models, ensuring that they are deployed efficiently and maintained effectively in production environments. In contrast, ML Engineering is primarily concerned with the

development and maintenance of an ML-based system. Thus MLOps emphasizes the operational aspects of machine learning, while AI Engineering is centered on the overall lifecycle of the system covering all system engineering concerns (from specification to maintenance) which includes MLOps for ML-based systems. MLOps involves collaboration between data scientists, ML engineers, and IT operations teams when AI Engineering involves system ad software engineers, data scientists, safety and cyber-security engineers. The end-to-end methodology (see Figure 4) supports all AI engineering activities where MLOps covers ML algorithm engineering and data engineering.

E. Deployment of a ML-component in the system

Deploying an ML component involves integrating it into an existing system, validating it and making it accessible for real-time or batch processing. The challenge for AI/ML-based systems lies in integrating, deploying and scaling a solution. The end-to-end methodology validates the quality of the data and knowledge, the process, and the added value delivered by AI/ML components, at a lower cost than the classical software/system engineering effort involved in automating and integrating a non-validated application with in-house and third-party systems. Combining the different engineering steps can require significant development effort, creating cost barriers when testing and validating ideas and prototypes that depend on integration with the rest of the system. Validating ML-based systems is more complex than manually coded systems. This is due to the behavior of ML-based systems, which depends heavily on data and knowledge, and for which models cannot be strongly specified a priori. Therefore, training data or knowledge-based models require qualification, similar to code [73].

Thus, verification testing of the ML-based system must also rely on the integration tests already performed at the "Integration of ML Component in System" step. They are usually run in a simulated environment (*i.e.*, with test benches or synthetic input data, or pre-recorded operational data). Consequently, testing the ML-based system consists again at least in regression testing (*i.e.*, test of no functional or non-functional regression), enabling to verify that the ML component features previously tested, integrated in the final ML-based system, still operate in conformance with their requirements based for example on

- Test by sampling and perturbation (empirical testing),
- Testing by formal verification of robustness (formal testing)...

However, it is also necessary to develop and run new verification tests, with potentially new tools and new test datasets, in order to ensure a complete requirements test coverage, prior to delivery to the Validation level.

F. Validation and Verification

Once the ML-based System has been successfully verified, it can be provided to the next engineering phase: the validation of the ML-based System. The difference between “verification” and “validation” is the following:

- Verification: evaluation against the system requirements that were written in order to design the ML-based System. Have we built the system right (as per design intent)?
- Validation: evaluation against the high-level needs identified during operational analysis. Have we built the right system (as per initial need)?

Then the "Validation, Qualification of ML-based System" engineering activity has to be performed:

- each time the “Operational Analysis” activity releases a new version of the operational/stakeholder needs (especially Intended Purpose Summary) against which the ML-based System shall be validated;
- and each time the “System Verification” activity releases a new version of verified ML-based System that is ready for validation.

Particular attention should be paid to stakeholders/operational requirements relating to the automation objectives that the system’s feature implemented by ML is intended to satisfy. Furthermore, ML-based system tests must be run in the target operational environment. It is no longer possible to simulate the environment using a test bench, synthetic input data or pre-recorded operational data.

The performance of a safety-critical system in its intended operational environment is a mandatory part of overall system validation. The process of traditional software validation involves establishing a chain of evidence that connects requirements to system-level tests. However, the use of machine learning techniques frustrates this approach due to the use of training data rather than a traditional design process. It is essential that software validation is based on tests that demonstrate a performance level commensurate with the criticality of the risks. These tests should be performed on a dataset that is fully representative of the factors that influence the model. The way in which the model’s functional characteristics and operational environment are specified may result in numerous factors influencing performance. To demonstrate the model’s effectiveness would require extensive testing datasets, potentially numbering in the millions of samples. Achieving this goal is an interesting prospect, but it is still at an early stage of research. Formal verification and simulation are interesting tracks to pursue. Therefore, verification requires ensuring that training and testing data cover all relevant operational conditions. In practice, this problem is generally made tractable by constraining the operational environment to a subset of all possible situations that could be dealt with by a human operator. The adoption of an ODD is the term given to that approach to limiting the operational needs of the system. Testing an ML component aims to detect discrepancies between the actual and intended behaviors of ML models. The term "ML testing" is used to describe any activity that is designed to reveal any bugs in ML items. "ML bugs" refer to any imperfection in an ML item that causes a discrepancy between the output.

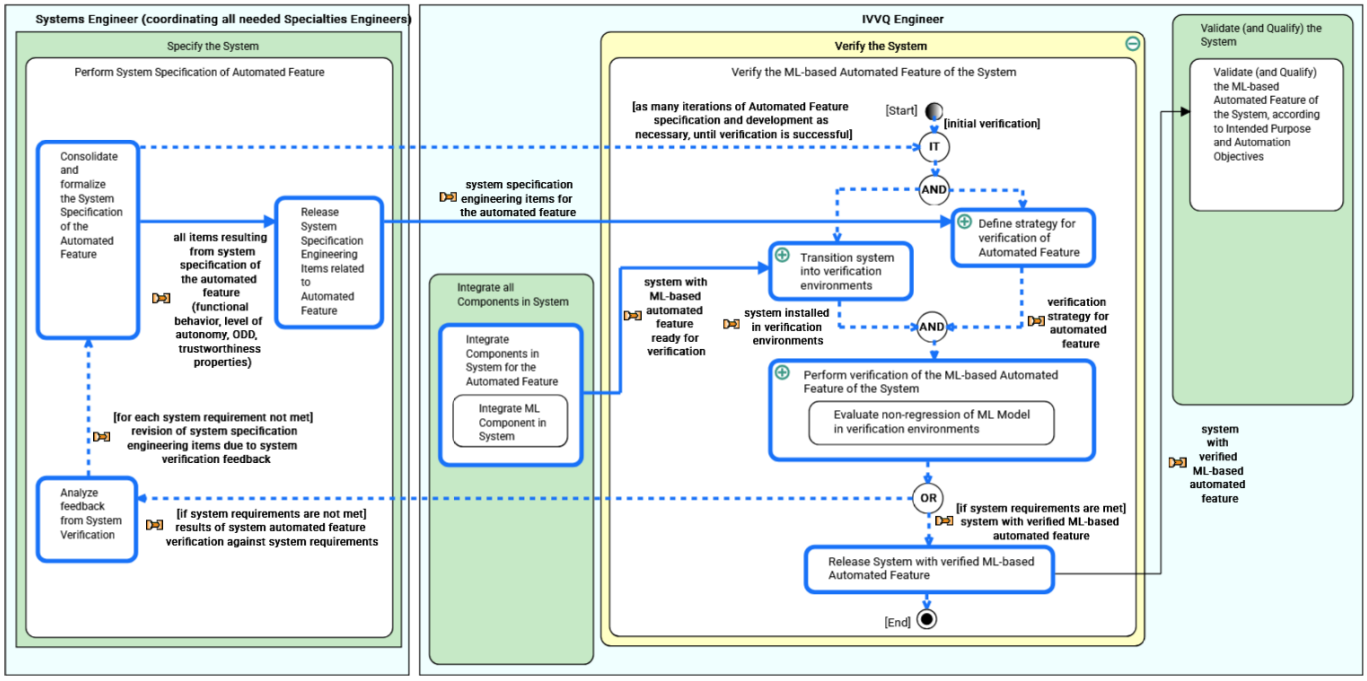

Figure 8. Verification Phase: verification of the ML-based feature of the system.

Testing an ML component aims to detect discrepancies between the actual and intended behaviors of ML models. Formal ML testing refers to any activity designed to reveal ML bugs, where an ML bug is defined as any imperfection in an ML item that causes a discrepancy between the model's output and the reference output. Examples of discrepancies could be due to a shift in the distribution of training and testing data, or an incorrect assessment of the suitability of the data for the task at hand; therefore, data is usually the cause of incorrect or unexpected errors.

This definition highlights three challenges to overcome. First, ML systems may have different types of "required conditions", *i.e.*, properties to verify. We can classify these as basic functional requirements (*e.g.*, correctness and model relevance) and non-functional requirements (*e.g.*, efficiency, robustness, fairness, interpretability). Different methods and metrics are required to verify such properties, so the selection of the best tools for verifying the component must be preceded by a definition of the required conditions: "What do we want to prove through testing?". Second, an ML bug may exist in the data, the learning program, or the framework. The testing strategy should address either the component itself or another "sub-component". This may make the testing more complex, since establishing a causal link between the bug and its source may be difficult, and defining a testing protocol allowing the distinction of independent and dependent variables is not trivial in an ML pipeline. Finally, testing activity may include several radically different approaches. These may include test input generation, test oracle identification, test adequacy evaluation, and bug triage. The selection of the approach must be based on a trade-off between the technical feasibility of performing such a test on the ML component and the required conditions initially formalized.

Quality control is an essential part of verifying and validating the ML component, and this can be achieved by estimating the success of the task solved by the component. Traditional metrics for regression problems include mean squared error (MSE) or mean absolute error (MAE), while classification problems can be evaluated using precision, accuracy, and recall. For classification problems, a confusion matrix depicting the distribution of true/false negatives/positives for each class is a practical tool for visualizing errors and allows most metrics to be computed (*e.g.*, precision, recall, sensitivity, specificity, F1 score, and ROC curve).

The most common evaluation protocol involves maintaining a hold-out validation set. This involves setting aside some of the data as the test set. The process involves training the model with the remaining data and tuning its parameters with the validation set, before finally evaluating its performance on the test set. The reason for splitting the data into three parts is to avoid information leaks. The main disadvantage of this method is that if a small amount of data is available, the validation and test sets will contain so few samples that tuning and evaluating the model will be ineffective. An alternative is k-fold cross-validation, which involves splitting the data into k partitions of equal size.

Another interesting approach is: Iterated k-fold validation with shuffling. This technique is useful when there are few data available and it is necessary to evaluate models as precisely as possible. Functional performance evaluation presents its own challenges. Selecting the most appropriate metrics to reflect the desired level of performance and choosing a suitable testing

protocol require careful consideration. However, the notion of quality control (QC) should go beyond simply estimating functional performance.

First, we note that the validation set is part of the ML algorithm design. The focus here is on the technical validity of the algorithm design, with few links to the operational constraints established in the specification phase. The influence of the training data is ignored at this stage, as traditional protocols do not necessarily take into account the informational value of the data points in each set. QC should encompass more than a simple evaluation of the ML algorithm. QC procedures should be formalized and deployed at each stage of the ML pipeline, with different objectives and verification strategies, but with one overarching objective: to ensure the quality of all processes involved in developing the ML component.

Although each domain has its own traditional ways of performing qualification (for example, data qualification has its own procedures), the link with the particularities and constraints of ML components is not always well established. Additionally, some aspects of verification and validation strategies are underestimated or not routinely considered in ML engineering. For example, data engineering information about the limits and constraints of the data should be reflected in the overall model evaluation strategy. The system in which the ML component is intended to operate must also provide its own set of constraints against which the component's compliance can be checked. This means that all parts of the ML pipeline should include specific QC procedures, and this information should be communicated to the relevant parts of the pipeline to inform the overall evaluation of the component's quality.

G. Maintenance and In-Service Support

Once the AI-based System has been validated and qualified, it can be deployed and declared in-service. Maintaining AI systems is a complex and evolving challenge. In many ways, this mirrors the rigorous and continuous effort. At its core, AI maintenance is not just about ensuring a model functions as intended at deployment; it is also about safeguarding its performance, reliability, and trustworthiness throughout its entire lifecycle. This is particularly important because AI systems are increasingly being integrated into high-stakes areas where failures can have severe or even catastrophic consequences. Maintaining AI systems is a complex and evolving challenge. In many ways, this mirrors the rigorous and continuous effort. At its core, AI maintenance is not just about ensuring a model functions as intended at deployment; it is also about safeguarding its performance, reliability, and trustworthiness throughout its entire lifecycle. This is particularly important because AI systems are increasingly being integrated into high-stakes areas where failures can have severe or even catastrophic consequences.

Robustness is a core concept in AI maintenance, referring to an AI system's ability to perform reliably in unexpected or adversarial conditions. Robustness is threatened by various factors during development and deployment. In development, the integrity of training data is crucial. Data can be compromised by noise from errors in data collection, annotation, or processing, or by data poisoning, where incorrect data is injected to degrade performance. This can lead to a model that performs well in testing but fails in real-world applications. Another threat is the backdoor attack, where an attacker embeds a hidden trigger in the model. When activated, the model's behavior can be manipulated without detection. These vulnerabilities are concerning in distributed learning environments, such as federated learning, where multiple parties contribute to the training process without full data visibility.

Once deployed, an AI model faces new challenges that can undermine its robustness. Adversarial examples, crafted inputs designed to deceive the model, exploit its sensitivity to small, often imperceptible, perturbations in the input data. An image recognition system might be misclassified due to a few pixels being altered in an image. Such vulnerabilities are dangerous in safety-critical applications like avionics, where a single misclassification can have life-or-death consequences. Another challenge in deployment is out-of-distribution generalization, the model's ability to handle inputs that differ from the data it was trained on. Real-world environments are dynamic, and data distributions can shift over time due to factors such as changing user behavior, sensor degradation, or evolving contextual factors. A model that performs well on its training data may struggle when faced with these shifts, leading to degraded performance or unexpected failures.

To address these challenges, AI maintenance has been proposed. It is a process akin to the maintenance of complex systems. An AI system requires regular monitoring and testing to remain reliable and maintain robustness. Inspection and diagnosis involve probing the model to identify vulnerabilities, anomalies, or degradation. Testing the model against adversarial examples is an example of monitoring for data drift. Soliciting feedback from users helps to understand the model's real-world behavior. Fixing and updating are next. This can be recalibrations or interventions like hardening the model against specific threats. Modules can be replaced if they are flawed. The cost and complexity of these activities depend on the identified issues and the requirements of the application.

An AI model inspector is a proactive framework that goes beyond the passive documentation provided by tools like model cards or data-sheets. Detect potential risks, such as back-doors, adversarial vulnerabilities, or data drift, and then take corrective actions to mitigate these risks. This could involve retraining the model on updated data, applying patches to address specific vulnerabilities, or even triggering a complete overhaul if the model's performance has degraded. The inspector framework represents a shift from reactive to proactive maintenance, where potential issues are identified and addressed before they lead to failures.

IV. Trustworthiness Attributes and Assessment

Trustworthiness is fundamental for the successful development and adoption of AI-based critical systems. Thus, trustworthiness assessment [76] can be defined as the process of evaluating and determining the level of trustworthiness of

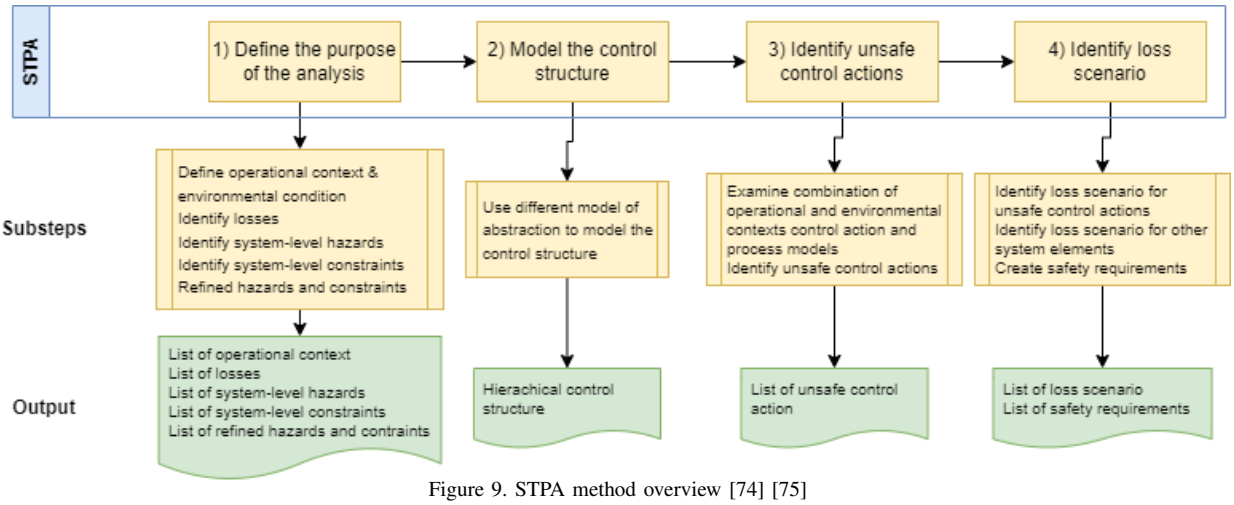

Figure 9. STPA method overview [74] [75]

a given characteristic, such as robustness [77] [78], accuracy, reliability [26], or effectiveness, in the context of AI systems engineering.

Nevertheless, it is very misleading to only judge how good an AI system is based on how accurate it is. It is also difficult to test and check the quality of software in the traditional way, and it is even difficult to measure test coverage at all. Trust and trustworthiness are complex, and so one of the main issues we face is to establish objective attributes such as accountability, accuracy, controllability, correctness, data quality, reliability, resilience, robustness, safety, security, transparency, explainability, fairness, privacy, and compliance with regulatory actors. We need to map these attributes onto the AI processes and its lifecycle and provide methods and tools to assess them. This highlights the importance of quality requirements, which are non-functional requirements and are particularly challenging in AI systems, although many of them can be considered in any critical system. Furthermore, this can also include risk and process considerations. The attributes and values for these requirements depend on things like how important the application is, what the AI system is used for, how it will be used, and the people involved. So, in some situations, some attributes may be more important than others, and new attributes may be added to the list [79]. Clear specifications of the non-functional requirements will help clarify these conflicts and can also encourage innovation that solves some of these conflicts, allowing us to fulfill more of them at the same time.

A. Risk analysis related to trustworthiness relationships between stakeholders

All interactions between the stakeholders (*e.g.*, engineers, operators, end-users, certification authorities, insurance companies, etc.) and the system are addressed by the trustworthiness relationships dimension.

Trustworthiness relationships must be established at each phase of the System lifecycle. They must also be maintained at each phase. This applies from engineering and design, until operation in a target environment. Indeed, during the engineering and design phases, engineers must be able to build trust on the system they will deliver to operators, which is an essential step in the process. Ultimately, operators must have confidence in the system features they will use.

The way in which trustworthiness relationships are established is dependent on the automation objectives that need to be achieved, as well as the environmental and human conditions that must be taken into account when operating in a trustworthy manner. As a result, trustworthiness relationships need to be analyzed from the viewpoint of each stakeholder involved in the automation Objectives, and defined and refined so that they can be supported by the system.

The dimension of trustworthiness relationships requires the application of an AI-specific risk analysis (see Figure 11). Indeed, due to the high level of uncertainty and unpredictability of the AI-based Automated Features outputs and behaviors, a new risk analysis approach related to the dimension of trustworthiness relationships is needed. Various techniques for hazard analysis such as Failure Modes and Effects Analysis (FMEA), Fault Tree Analysis (FTA), Hazard and Operability Analysis (HAZOP), System Theoretic Accident model and Processes (STAMP) and System Theoretic Process Analysis (STPA) are common. The STAMP framework is an accident causality model that provides a new paradigm for STPA.based system safety engineering.

In our conte)xt, the Confiance.ai research program has proposed a methodological process. The process relies on both the unified approach for trustworthiness assessment defined in our previous work [29] and the STPA method (see Figure 9), which is identified as relevant for analysis purposes.

STPA [75] is a system approach that considers potential dysfunctional system's characteristics and behaviors as a system control problem and not only as a problem of component failure. It does not replace traditional failure analysis approaches but complements them. In *Confiance.ai*, STPA is extended beyond its traditional safety analysis domain to trustworthiness characteristics/properties risks analysis and control, to be applied for each autonomy objective and feature defined in the

Concept name	Definition
Misuse	[ISO 21448: 2021] usage of the system by a human in a way not intended by the manufacturer or the service provider
Error	[ARP 4754 A] A mistake made by a crew member or maintenance person, or a mistake in the requirements, design or implementation (derived from AMC 25.1309). [ISO 26262: 2011] A discrepancy is when a value or condition is not the same as the true value. It could be a computed value. Or an observed value. Or a measured value. Or a theoretically correct value. Note 1: An error can arise as a result of unforeseen operating conditions or due to a fault within the system, subsystem or component being considered. Note 2: A fault can manifest itself as an error within the considered element and the error can ultimately cause a failure.
Failure	An occurrence, which affects the operation of a component, part or element such that it can no longer function as intended, (this includes both loss of function and malfunction). Note: errors may cause Failures, but are not considered Failures. (AMC 25.1309)
Hazard	Definition from STPA: A system state or set of conditions that, together with a particular set of worst-case environmental conditions, will lead to an accident (loss). [ARP 4754 A] Extended for trustworthiness: A condition resulting from failures, external events, errors or a combination of these factors affecting trustworthiness.
Worst-case environmental condition	Environmental, non-controllable context
Risk	[ARP 4754 A] The level of severity of an occurrence is dependent on its frequency (probability).
Accident-Loss	Definition from STPA, extended for trustworthiness: An undesired or unplanned event that results in a loss, including loss of human life or human injury, property damage, environmental pollution, mission loss, trustworthiness loss etc. Definition from STAMP: An undesired or unplanned event that causes loss, damage, or injury [80].
Mitigation	Any means enabling risk reduction (occurrence likelihood and/or impact with barriers) at any step of the System of interest lifecycle (*e.g.*, specification, design, training...).
Loss/Damage	[STPA Handbook] extended for trustworthiness: A loss involves the loss of something of value to stakeholders. They may consider this to include anything from loss of human life or injury to property damage, environmental pollution, loss of mission, loss of reputation, loss or leak of sensitive information or loss of trustworthiness.

Figure 10. Safety Analysis concept definition [29]

ODD. Therefore, STPA can be applied to analyze and mitigate risk of trustworthiness loss.

Remind that STPA is a system-theoretic safety analysis method designed to identify and mitigate risks in complex systems by focusing on control structures and unsafe interactions rather than just component failures. It is particularly useful for autonomous, cyber-physical, or human-machine systems where traditional hazard analysis may fall short. The process is structured into four key steps (see Figure 9), each building on the previous one to ensure traceability from high-level losses to specific scenarios:

1) Identify accidents and hazards list, and associated unacceptable losses related to the System of Interest (either material losses or immaterial losses, *e.g.,* mission, trustworthiness loss. Identify also the boundary of the analysis and defines those losses and hazards that must be prevented as well as high-level operational and system constraints/requirements needed to prevent them.
2) Identify and model the control structures, starting from the operational environment, and refining through the abstraction layers of the system analysis (*i.e*, System level, Architectural level, AI Component level)
3) Identify Unsafe Control Actions (UCAs) leading to hazards and losses, and specify requirements and constraints to prevent them.
4) Identify scenarios leading to UCAs or hazards, and specify requirements to mitigate the risks."

To define the process of trustworthiness risk analysis, traditional Safety Analysis concepts of the aeronautical business domain have been considered (*e.g.*, failure). Other business domains (*e.g.*, railway, automotive...) could introduce their dedicated concepts (see Figure 10). The resulting process is described in Figure 11, as a pattern that needs to be iteratively applied at several steps of the end-to-end method.

The principle is that a specific Trustworthiness Engineering team, once it has analyzed the trustworthiness properties, has also to analyze the trustworthiness loss risks, according to the approach AI specific risk analysis approach. It has also to perform potential traditional risks analysis (*e.g.*, Safety failure modes analysis approaches). In addition to the system specification including Operational Design Domain (ODD) analysis, and trustworthiness assessment processes, a risk analysis process is essential to address and mitigate the risks related to AI technologies, based on add-hoc control-structures specifications.

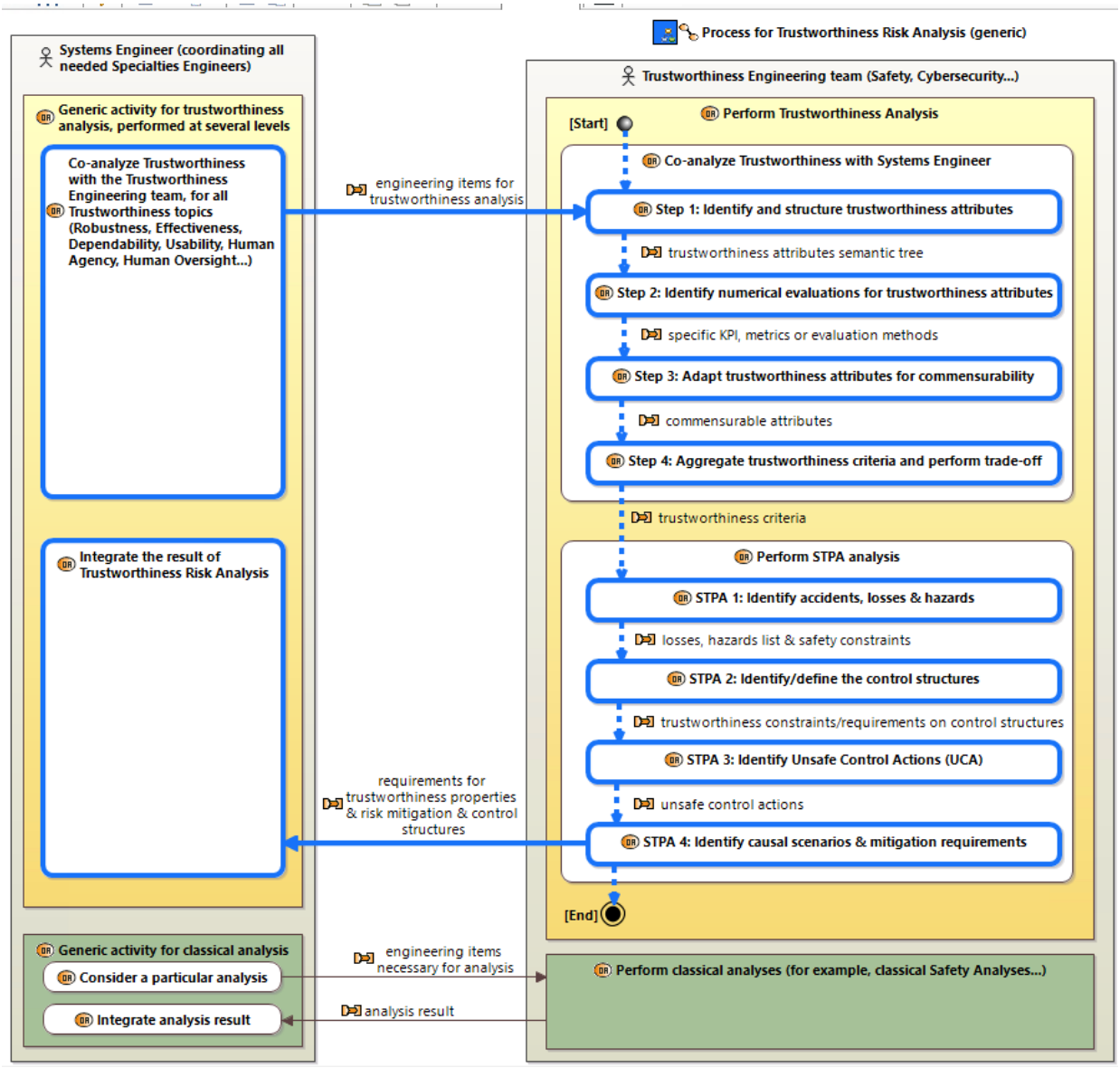

Figure 11. Trustworthiness Risk Analysis - Generic process

B. *Local trustworthiness assessment*

Thus, by leveraging system engineering best-practices, ML development workflows, and testing procedures, the end-to-end methodology ensures that trustworthiness attributes are embedded in every stage of the AI system life-cycle, from conception to maintenance. The Confiance.ai framework focuses on the following attributes:

- **Robustness** covering **Safety and Security**: High-risk systems must be as resilient as possible against errors, faults or inconsistencies that may occur within the system or the environment in which it operates, and also against attempts by unauthorized third parties to alter its use, outputs or performance by exploiting system vulnerabilities; furthermore, the technical solutions aiming to ensure the cybersecurity of high-risk AI systems must be appropriate to the risks and circumstances. Various perturbations (*i.e.*, variations in input data and operating conditions) should not be an issue for robust AI systems. Therefore, an AI-based system must meet rigorous safety and security requirements [81] (see Figure 12):
  - Safety analysis and certification based on standards.
  - Cybersecurity counter-measures, integrated on the AI pipeline.

  This requires :
  - Adversarial robustness, ensuring the system is not easily manipulable by adversarial attacks.
  - OOD Robustness (Out-Of Distribution), the system must generalize well across different environment and be trained on diverse datasets.
  - model monitoring, ensuring a continuous evaluation

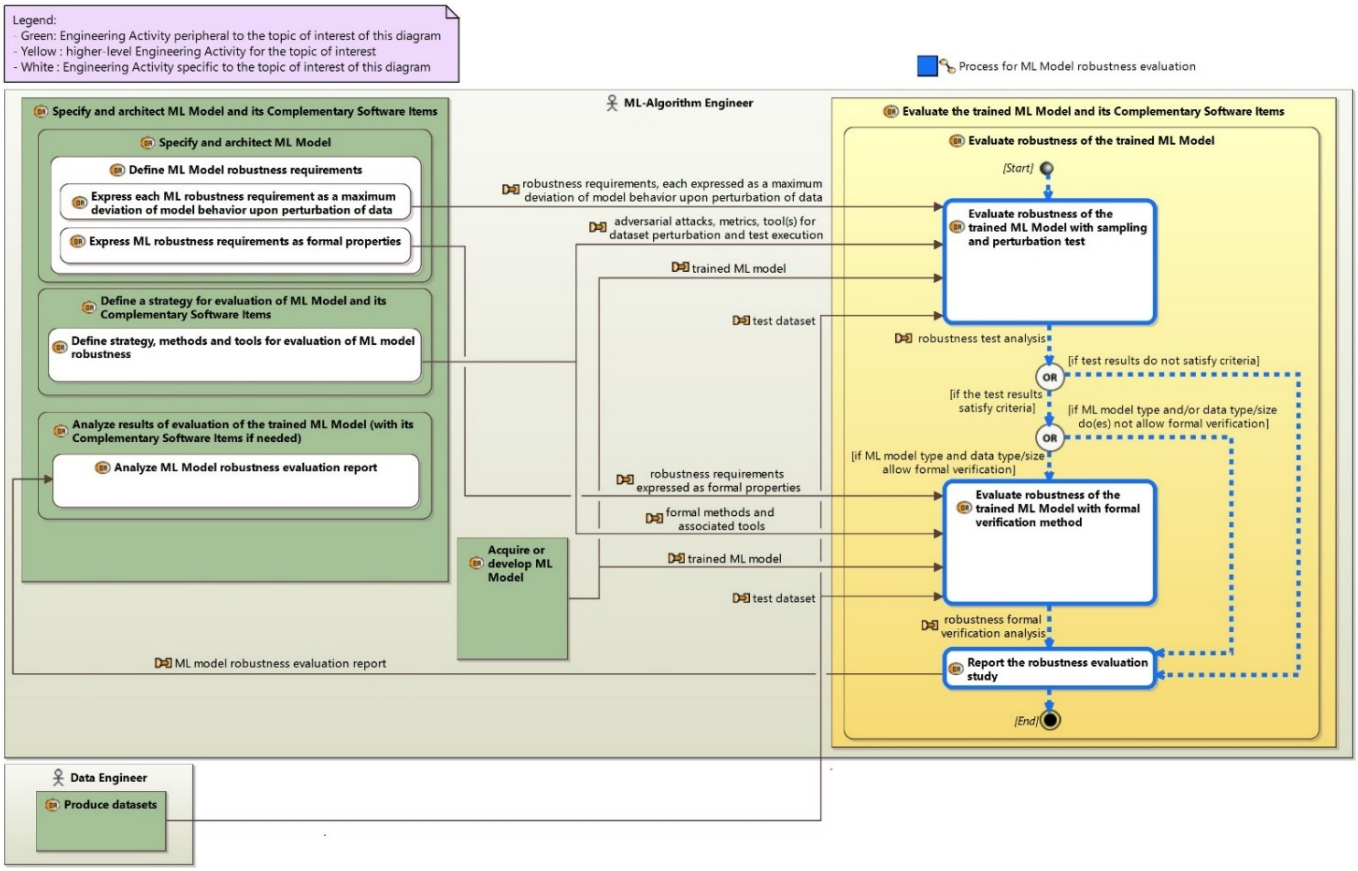

Figure 12. ML model robustness evaluation process considering two complementary strategies

of the AI models, to detect performance degradation.

To evaluate the robustness of ML model, the end-to-end methodology proposes a strategy made of two successive phases:

- Robustness test by sampling and perturbation (empirical evaluation).
- Formal verification of robustness (formal evaluation);

The test-based phase consists in comparing, on one hand, the behavior of the ML model fed with a perturbed dataset, and on the other hand, its nominal behavior.

The formal evaluation-based phase uses formal methods and tools to verify one or several mathematical properties (here, related to robustness) of the ML model. Ideally, such a property shall be formally verified on the whole Model, whatever the input data. However, practically, because of constraints on the formal verification tools, the property is formally verified only for given input data: it proves that the ML model is locally robust, at given points of the test dataset. Each phase implies a specific expression of ML model robustness requirements.

The first phase is relatively inexpensive, compared to the second one. By testing the model with different inputs and perturbations, information is obtained about the performance of the model in different scenarios and its ability to generalize well despite data perturbation. However, this type of evaluation has limited confidence because it only tests a subset of possible scenarios but it may not uncover all potential issues or weaknesses.

On the other hand, the second phase involves rigorous mathematical analysis of the model robustness: it is more expensive and time-consuming compared to the first phase. Formal verification provides a higher level of confidence in the model's performance because it is based on sound proofs. However, it is important to note that formal verification may not be possible for certain types of models due to their complexity or lack of formal specifications. In this sense, the adoption of a formal verification to evaluate the robustness of ML models depends on certain constraints such as the acceptability of formal proofs, the compatibility of the verification tools with the ML model algorithm, and the dimension of the data space.

The interest of starting with sampling and perturbation test is to quickly identify any major issues or weaknesses in the model. If the model performs well in this step, it can then be subjected to formal

verification to obtain a more expensive but also more reliable result.

This combination of approaches allows for the most comprehensive evaluation of the model robustness, considering both cost-effectiveness and confidence.

- **Transparency, Explainability, Interpretability and Comprehensibility**. The principle of "having a human in the loop" is at the core of responsible and trustworthy AI. Transparency is vital for effective human control and oversight, including instructions for safe use and information about the level of accuracy, robustness, and cybersecurity of the critical AI system. This enables us to: (i) Properly understand the relevant capacities and limitations of the system and monitor its operations, including in view of detecting and addressing anomalies, dysfunctions and unexpected performance; (ii) Maintain awareness of the potential tendency to automatically rely or over-rely on the output produced by the system; (iii) Accurately interpret the system's output; and (iv) Decide not to use the system or otherwise disregard, override or reverse the system's output. Moreover, trustworthy AI should be transparent and its decisions should be interpretable where
  - Explainability deals with the capability to provide the human with relevant information on how an AI application is coming to its result.
  - Interpretability relates to the capability of an element representation (an object, a relation, a property...) to be associated with the mental model of a human being. It is a basic requirement for an explanation.
  - Comprehensibility refers to the capability of an element representation (an object, a relation, a property...) to be understood by a person according to its level of expertise or background knowledge.

  This requires:
  - Post-hoc explainability tools, to provide insights into model decisions.
  - model simplification strategies to enhance interpretability.
  - Human-in-the-loop validation to ensure AI decisions align with expert knowledge.

  There is a profusion of methods, tools, and solutions available, each with its own set of advantages, drawbacks, and trade-offs [84]. The many different approaches show how tricky it is to make sure that AI and machine learning models can explain their predictions and decisions. Choosing the right way to make models explainable is a technical and strategic decision. It depends on the unique needs and limits of the people it will be used by, the specific example it will be used for, and the wider situation in which the AI system will be used. What works for a medical diagnosis model may not work for the aeronautic domain, and what regulators expect can be very different from what end-users or business stakeholders expect. The Confiance.ai program provides a "Methodological Guideline for Explainability" (https://catalog.Confiance.ai/) which is designed to be a complete guide to help people use AI. It will explain why explainability is important, highlight the many available methods, and offer guidance on selecting the most suitable approach based on the specific situation.
- **Fairness and Bias Mitigation**. A key concern with data-driven AI (such as ML) is the amplification of biases. Therefore, we have first to take appropriate measures to detect, prevent and mitigate possible biases and to use high-quality datasets for training, validation, and testing, as the output of the AI system depends largely on the quality of the training data. The data must be relevant, sufficiently representative and, to the best extent possible, free of errors and complete. AI models should be free from discriminatory biases. This involves:
  - Bias detection and correction techniques, in the data processing and model training phases.
  - Regulatory alignment with fairness standards.

The end-to-end methodology integrates those attributes throughout the AI system life-cycle, namely in:

- **Operational Design Domain (ODD) definition**: Critical AI systems are subject to rigorous regulatory requirements, including conformity assessments and post-market surveillance. The EU AI Act establishes a risk classification system for AI systems based on their intended purpose. This means that the use for which an AI system is intended by the provider, including the specific context and conditions of use, determines its risk classification. This ensures that regulatory scrutiny aligns with the system's anticipated function and impact.
  - Define the operational boundaries where the AI system is expected to function reliably.
  - Establish clear environmental constraints for the AI-system's development.

  The ODD is a description of measurable foreseeable operating conditions within which a system/component shall operate. A traceability property shall be assured between the different levels of ODD (system, subsystem or component).
- **Systems Engineering**
  - Ensure AI system-level requirements are defined in alignment with overall system objectives.
  - Align AI-based system requirements with preexisting system engineering standards and certification guidelines.
- **Data Engineering and Data Quality Assessment**
  - Rely on a robust data pipeline to guarantee data integrity, consistency, and traceability across the engineering cycle.
  - Implement bias mitigation strategies at the data collection and processing stages.
  - Use adaptive data augmentation strategies to improve data diversity and model generalization to distribution shifts and operational scenarios.

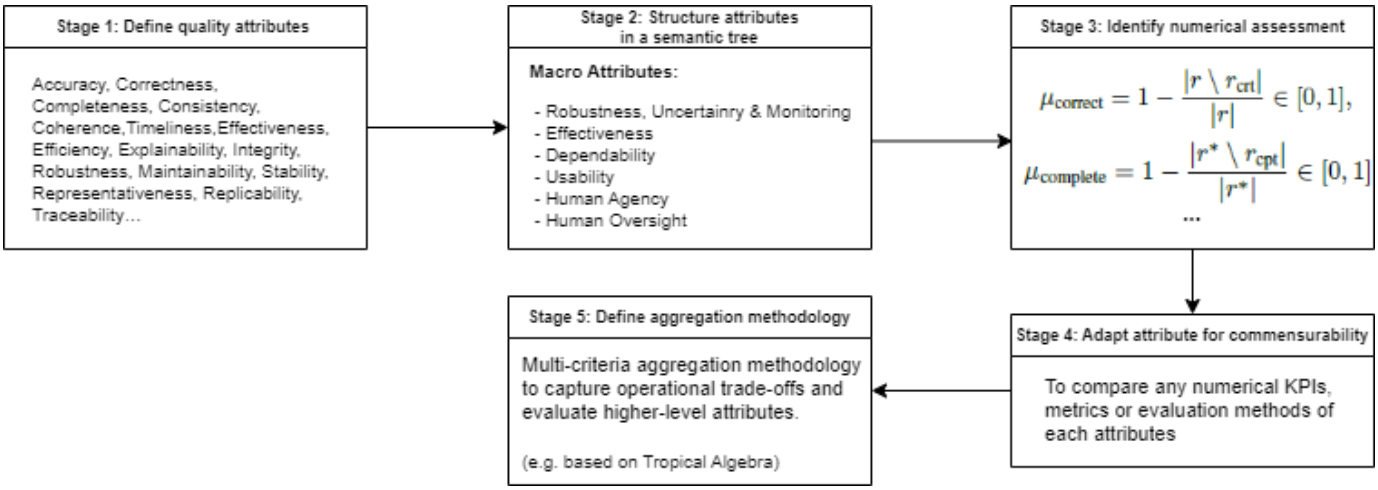

Figure 13. The unified approach based on MCDA [82] [83]

- **ML Algorithm Engineering**
  - Use ML robustness techniques, designed to handle perturbation and adversarial outputs.
  - Incorporate explanability techniques to have understandable decisions.
  - Apply Uncertainty quantification techniques to asses the model's confidence.
- **Verification and Validation**
  - Perform extensive simulation-based testing to asses performances under edge cases.

In addition, measuring how trustworthy AI systems are is tricky. The ideas behind them are complicated, the characteristics they produce are different, and you can't always compare them. The Confiance.ai program proposes an innovative way to measure trustworthiness using (max,+) algebra [85] based on a complete hierarchical model that brings together different properties, such as how strong, effective, dependable, easy to use and human agency, and human oversight) into a single assessment method. This offers advantages over traditional weighted averaging methods by better handling extreme values and preserving sensitivity to critical indicators, while maintaining sensitivity to critical indicators to provide detailed, understandable assessments of AI-based system trustworthiness.

C. Global Trustworthiness Assessment

As it is not straightforward to select the relevant attributes for assessing AI trustworthiness, given that the choice depends on the context of application. This context is modeled according to a number of elements, including the Operational Design Domain (ODD), the intended domain of use, the nature and roles of the stakeholders, and so on. The attributes may be quantitative, typically comprising numerical values derived from measurements or providing a comprehensive statistical overview of a phenomenon. Alternatively, they may be qualitative, based on the detailed analysis and interpretation of a limited number of samples. Then, the second activity mentioned above on the characterization of the trustworthiness evaluation is broken down into several activities, according to the Multi-Criteria Decision Aiding (MCDA) method [82] [83]. Those are:

1) *Define trustworthiness characteristics*. All the characteristics of the considered item are identified and described (*i.e.*, their name, properties).
2) *Structure attributes in a semantic tree*. Characteristics (*i.e.*, quality attributes) are organized in a tree, from the most general down to the leaf characteristics.
3) *Identify numerical evaluations*. Each characteristic is typed by a numerical value domain.
4) *Adapt attribute for commensurability*. Characteristics can follow different forms of distribution with different value domains. The purpose is to make them compatible so that they can be compared and operated together.
5) *Define the aggregation methodology*. MCDA enables one to explore several solutions, compare them, and to keep the best one.

Once the list of relevant attributes has been defined, the aggregation of several attributes remains complex due to issues of commensurability. This is because the attributes in question are not of the same unit; for example, combining "*oranges and apples*" is not a meaningful exercise. Furthermore, it is necessary to make compromises and arbitrate between the attributes. This means that the value of each attribute must be transformed into a scale that is consistent across all attributes and reflects the preferences of a stakeholder. Furthermore, the values assigned to the scales for the various criteria must be aggregated. These elements represent the primary stages of a problem-solving process that employs an Multi-Criteria Decision Aiding (MCDA) approach [85].

MCDA is a generic term for a collection of systematic approaches developed specifically to assist one or several decision makers in assessing or comparing alternatives based on multiple criteria [86]. The challenge lies in the fact that the decision-making criteria are often numerous, interdependent, and occasionally in conflict with each other. For instance, there

may be a conflict between effectiveness and other criteria, such as robustness, explainability, or affordability. The viewpoints are quantified through the use of attributes. Aggregation functions are frequently used to facilitate comparisons between alternatives evaluated on the basis of multiple, potentially conflicting criteria. This is achieved by synthesizing their respective performances into overall utility values. These functions must be sufficiently expressive to align with the preferences of the stakeholders involved, enabling the identification of the most preferred alternative or facilitating the negotiation of compromises among the criteria. It is important to note that improving one criterion may necessitate a trade-off in another.

V. Conclusion and future works

The Confiance.ai program has evolved since its kick-off in 2021, with a first year dedicated to covering the academic and industrial state of the art related to ML-based system design. Subsequent years (2022-2023) were dedicated to the accurate characterization of industrial use cases, the development and evaluation of technological components to address specific aspects of reliability, and the construction of an end-to-end method revisiting all stages of the engineering cycle for the design, integration, and evaluation of ML components [9]. The last year (2024) encompasses the evaluation of this end-to-end method, the completion and dissemination of key results, and the guarantee of their continuation and sustainability under the aegis of a new research initiative currently under construction. To facilitate the adoption of the tool-based methodology by industry, several implementations of the 2023 version have been carried out on use cases.

These experiments have demonstrated the importance of integrating diverse tools and methods to address expectations regarding trusted ownership, as illustrated by the following two examples: In a use case involving autonomous driving, the analysis of dataset diversity reveals a limited presence of night-time images, prompting the generation of synthetic night-time data. This data exhibits a "domain gap" and undergoes "domain adaptation" prior to integration into the model training data. These tools, instrumental in the construction of datasets, will also be reused in the supervision stage of the use case. In an aeronautical use case called LARD for "Landing Approach Runway Detection" [87] and represented Figure 14, a data quality supervision module is incorporated to consolidate the confidence score of an ML model (see Figure 14). In this example, local image quality estimators (*e.g.*, level of blur, brightness) are taken into account in the detection zone of the landing strip that is being detected. The combination of these indicators with the other indicators intrinsic to the model facilitates the establishment of a level of confidence for the system component. In addition to providing a numerical value, this implementation serves as a tool to facilitate the interpretation of model and data errors.

The Confiance.ai program is opening up two major outcomes to the community as a "digital common good". First, it provides a body of knowledge describing an end-to-end method of AI engineering. This makes it possible to characterize and qualify the trustworthiness of a data-driven AI system and integrate it into industrial products and services. Second, this method is applicable to any sector of activity. A catalog of developed and/or mature technological components to increase the level of trust in AI integrated into critical systems.

The Body of Knowledge (BoK) is one of the main outcomes because it provides access to a navigable version of this end-to-end methodology that covers the activities structuring the engineering cycle of a critical system based on ML (https://bok.Confiance.ai/). This compendium of expertise from multiple disciplines is a corpus that articulates the system level with the model and data levels in the engineering process. It is continuously updated and expanded and is expected to continue beyond the program. The content provided in the body of knowledge is structured with an end-to-end engineering method in mind and can be navigated through different roles in this process, namely through the field of application of different engineering profiles: These roles include, but are not limited to, the following: machine learning algorithm engineer, data engineer, embedded software engineer, IVVQ (Integration, Validation, Verification and Qualification) engineer or system engineer.

The following simplified high-level view of the BoK is presented as a gateway to the end-to-end method for engineering trustworthy ML-based systems.The body of knowledge presents the stages of the methodology, from operational analysis and specification of the function of the system that one wishes to automate through the use of ML technology, to verification/validation/qualification, including the development and implementation of the ML model. The navigation through each stage and according to each role facilitates the visualization of the activities, sub-activities and workflow to be carried out when developing a reliable ML-based system.This corpus is thus a compendium of expertise from multiple disciplines because it links the system level with the model and data levels in the engineering process.It is continuously updated and expanded, and this is planned beyond the program.

The catalog (https://catalog.Confiance.ai/) is a web application that allows users to consult the results of the Confiance.ai program. It employs filtering and search functions (sorting, categories, etc.) to facilitate navigation through the various results, which can be either documents or software. Results categorized as "documentary" are exclusively of a literary nature, including reports (studies or benchmarks), state of the art, doctoral theses or good practice guides. "Software" results are components intended to be run directly or through another application, such as a web application, a library, a plugin or a binary executable.

Acknowledgment

This work has been supported by the French government under the "France 2030" program, as part of the SystemX Technological Research Institute within the Confiance.ai Program (www.Confiance.ai) and the *CSIA* (Confiance dans les Systèmes d'IA) project.

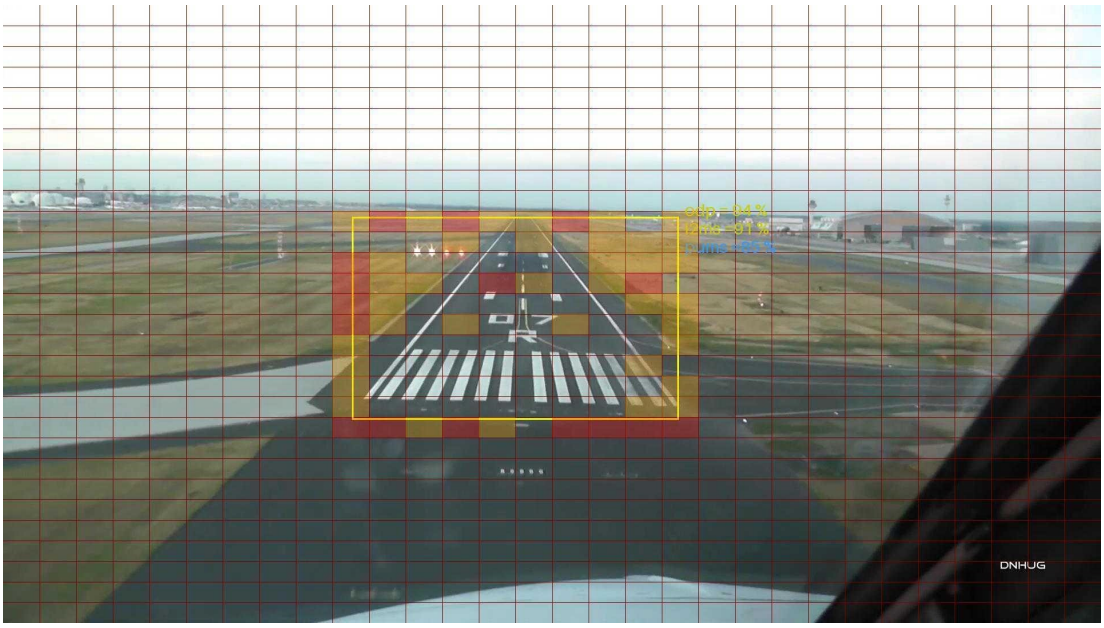

Figure 14. Example of the implementation of a supervision tool on the LARD [87] use-case

REFERENCES

[1] K. Quintero et al., "An end-to-end method for operationalizing trustwothiness in AI-based critical systems", in *15th International Conference on Performance, Safety and Robustness in Complex Systems and Applications PESARO 2025*, 2025.

[2] European Commission, *Proposal for a Regulation of the European Parliament and of the Council laying down Harmonised Rules on Artificial Intelligence (Artificial Intelligence Act) and Amending Certain Union Legislative Acts*, 2021.

[3] M. Felderer and R. Ramler, "Quality Assurance for AI-Based Systems: Overview and Challenges (Introduction to Interactive Session)", in *International Conference on Software Quality*, Springer, 2021, pp. 33–42.

[0] ISO/IEC 25024:2015, *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Measurement of data quality*, 2015.

[4] H. Liu et al., "Trustworthy AI: A computational perspective", *ACM Transactions on Intelligent Systems and Technology*, vol. 14, pp. 1–59, 2022.

[5] HLEG, *A definition of AI: Main capabilities and scientific disciplines*, Definition developed for the purpose of the deliverables of the High-Level Expert Group on AI, 2018.

[6] A. Horneman et al., *AI Engineering: 11 Foundational Practices-Recommendations for Decision Makers from Experts in Software Engineering, Cybersecurity, and applied Artificial Intelligence White Paper DM19-0624, 06.06*. Carnegie Mellon University, Software Engineering Institute (SEI), 2019.

[7] M. Gonzalez et al., "Introducing RUM: A Methodological Contribution for Engineering Trustworthy AI Components in Industrial Systems", in *Proceedings of the AAAI Symposium Series*, vol. 7, 2025, pp. 153–160.

[8] M. Adedjouma et al., "Engineering dependable AI systems", in *17th IEEE Annual System of Systems Engineering Conference (SOSE)*, 2022, pp. 458–463.

[9] A. Awadid et al., "Ensuring the reliability of AI systems through methodological processes", in *2024 IEEE 24th International Conference on Software Quality, Reliability and Security (QRS)*, IEEE, 2024, pp. 139–146.

[10] A. Avizienis et al., "Basic concepts and taxonomy of dependable and secure computing", *IEEE Transactions on Dependable and Secure Computing*, vol. 1, pp. 11–33, 2004.

[11] J. Cho et al., "Stram: Measuring the trustworthiness of computer-based systems", *ACM Computing Surveys (CSUR)*, vol. 51, pp. 1–47, 2019.

[12] NSTC, *The National Artificial Intelligence Research and Development Strategic Plan: 2019 Update*. National Science and Technology Council (US), 2019.

[13] E. Schmidt et al., "National security commission on artificial intelligence (AI)", National Security Commission on Artificial Intelligence, Tech. Rep., 2021.

[14] UNESCO, *Recommendation on the Ethics of Artificial Intelligence*, 2022. Accessed: Nov. 8, 2024. [Online]. Available: https://unesdoc.unesco.org/ark:/48223/pf0000381137.

[15] OECD, *Recommendation of the Council on Artificial Intelligence*, Legal Instruments, May 2019. Accessed: Nov. 3, 2024. [Online]. Available: https://legalinstruments.oecd.org/en/instruments/OECD-LEGAL-0449.

[16] OCDE, *G7 Hiroshima Process on Generative Artificial Intelligence (AI)*, 2023. DOI: https://doi.org/https://doi.org/10.1787/bf3c0c60-en. [Online]. Available: https://www.oecd-ilibrary.org/content/publication/bf3c0c60-en.

[17] EASA, *Concept Paper First Usable Guidance for Level 1 Machine Learning Applications*, 2021. [Online]. Available: https://www.easa.europa.eu/en/easa-concept-paper-first-usable-guidance-level-1-machine-learning-applications-proposed-issue-01pdf.

[18] P. Ala-Pietilä et al., *The Assessment List for Trustworthy Artificial Intelligence (ALTAI)*. European Commission, 2020.

[19] M. Mock et al., *Management system support for trustworthy artificial intelligence*, 2021.

[20] ISO/IEC DIS 42001, *Information technology — Artificial intelligence — Management system*, 2022.

[21] B. Stanton et al., "Trust and artificial intelligence", *NIST preprint*, vol. 10, 2021.

[22] IEEE 7000, *IEEE Standard Model Process for Addressing Ethical Concerns during System Design*, 2021.

[23] ETSI, *Securing Artificial Intelligence (SAI); Mitigation Strategy Report*, 2021.

[24] J. Mattioli et al., "Empowering the trustworthiness of ML-based critical systems through engineering activities", *arXiv preprint arXiv:2209.15438*, 2022.

[25] B. Braunschweig et al., "The wall of safety for AI: Approaches in the confiance.ai program", in *Workshop on Artificial Intelligence Safety (SAFEAI)*, 2022.

[26] J. Mattioli et al., "AI engineering to deploy reliable AI in industry", in *2023 Fifth International Conference on Transdisciplinary AI (TransAI)*, 2023, pp. 228–231.

[27] R. Gelin, "Confiance.ai program software engineering for a trustworthy AI", in *Producing Artificial Intelligent Systems: The Roles of Benchmarking, Standardisation and Certification*, Springer, 2024, pp. 11–29.

[28] A. Awadid et al., "AI Systems Trustworthiness Assessment: State of the Art", in *Workshop on Model-based System Engineering and Artificial Intelligence-MBSE-AI Integration 2024*, 2024.

[29] A. Awadid et al., "Ensuring the reliability of AI systems through methodological processes", in *2024 IEEE 24th International Conference on Software Quality, Reliability and Security (QRS)*, 2024, pp. 139–146.

[30] A. Awadid, B. Robert, and B. Langlois, "Mbse to support engineering of trustworthy AI-based critical systems", in *12th International Conference on Model-Based Software and Systems Engineering*, 2024.

[31] A. Awadid et al., "Towards engineering processes to guide the development of trustworthy ML systems", in *2024 IEEE International Symposium on Systems Engineering (ISSE)*, IEEE, 2024, pp. 1–6.

[32] V. Liubchenko, "Specific aspects of software development process for ai/ml-based systems", in *2022 IEEE 17th International Conference on Computer Sciences and Information Technologies (CSIT)*, IEEE, 2022, pp. 470–473.

[33] P. Koopman, F. Fratrik, et al., "How many operational design domains, objects, and events?", *Safeai@ aaai*, vol. 4, no. 4, 2019.

[34] F. Pedregosa et al., "Scikit-learn: Machine learning in python", *the Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[35] J. Mattioli et al., "Information quality: The cornerstone for ai-based industry 4.0", *Procedia Computer Science*, vol. 201, pp. 453–460, 2022.

[36] L. Mattioli et al., "Data curation matters: Model collapse and spurious shift performance prediction from training on uncurated text embeddings", *arXiv preprint arXiv:2506.17989*, 2025.

[37] M. Mazumder et al., "Dataperf: Benchmarks for data-centric AI development", *arXiv preprint arXiv:2207.10062*, 2022.

[38] J. Jakubik et al., "Data-centric artificial intelligence", *arXiv preprint arXiv:2212.11854*, 2022.

[39] M. Jarrahi, A. Memariani, and S. Guha, "The principles of data-centric AI (DCAI)", *arXiv preprint arXiv:2211.14611*, 2022.

[40] G. Mountrakis and B. Xi, "Assessing reference dataset representativeness through confidence metrics based on information density", *ISPRS journal of photogrammetry and remote sensing*, vol. 78, pp. 129–147, 2013.

[41] H. Delseny et al., "White paper machine learning in certified systems", *arXiv preprint arXiv:2103.10529*, 2021.

[42] Z. Gong et al., "Diversity in machine learning", *IEEE Access*, vol. 7, pp. 64 323–64 350, 2019.

[43] M. Dereziński, "Fast determinantal point processes via distortion-free intermediate sampling", in *Conference on Learning Theory*, PMLR, 2019, pp. 1029–1049.

[44] Z. Gong et al., "Diversity-promoting deep structural metric learning for remote sensing scene classification", *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, pp. 371–390, 2017.

[45] C. Zhang et al., "Active mini-batch sampling using repulsive point processes", in *Proceedings of the AAAI conference on Artificial Intelligence*, vol. 33, 2019, pp. 5741–5748.

[46] Y. LeCun et al., "Gradient-based learning applied to document recognition", *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, 1998.

[47] V. Gurupur and M. Shelleh, "Machine learning analysis for data incompleteness (MADI): Analyzing the data completeness of patient records using a random variable approach to predict the incompleteness of electronic health records", *IEEE Access*, vol. 9, pp. 95 994–96 001, 2021.

[48] C. Holden et al., "The electronic health record system and hospital length of stay in patients admitted with hip fracture", *Am J Research Nurs [Internet]*, pp. 1–5, 2015.

[49] Y. Teo et al., "Benchmarking quantum tomography completeness and fidelity with machine learning", *New Journal of Physics*, vol. 23, p. 103 021, 2021.

[50] C. Tran, *Evolutionary machine learning for classification with incomplete data*, 2018.

[51] B. Schouten et al., "Indicators for the representativeness of survey response", *Survey Methodology*, vol. 35, pp. 101–113, 2009.

[52] M. Berkesewicz, "A two-step procedure to measure representativeness of internet data sources", *International Statistical Review*, vol. 85, pp. 473–493, 2017.

[53] M. Blatchford et al., "Determining representative sample size for validation of continuous, large continental remote sensing data", *International Journal of Applied Earth Observation and Geoinformation*, vol. 94, p. 102 235, 2021.

[54] M. Qi et al., "Quantifying representativeness in randomized clinical trials using machine learning fairness metrics", *JAMIA open*, vol. 3, ooab077, 2021.

[55] Y. Geifman and R. El-Yaniv, "Selective classification for deep neural networks", *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[56] Y. Geifman and R. El-Yaniv, "Selectivenet: A deep neural network with an integrated reject option", in *International Conference on Machine Learning*, 2019, pp. 2151–2159.

[57] A. Krizhevsky, G. Hinton, et al., *Learning multiple layers of features from tiny images*, 2009.

[58] K. Pei et al., "Deepxplore: Automated whitebox testing of deep learning systems", in *proceedings of the 26th Symposium on Operating Systems Principles*, 2017, pp. 1–18.

[59] Y. Tian et al., "Deeptest: Automated testing of deep-neural-network-driven autonomous cars", in *Proceedings of the 40th International Conference on Software Engineering*, 2018, pp. 303–314.

[60] A. Odena et al., "Tensorfuzz: Debugging neural networks with coverage-guided fuzzing", in *International Conference on Machine Learning*, PMLR, 2019, pp. 4901–4911.

[61] F. Adjed et al., "Coupling algebraic topology theory, formal methods and safety requirements toward a new coverage metric for artificial intelligence models", *Neural Computing and Applications*, vol. 34, pp. 1–16, 2022.

[62] J. Bolte et al., "Towards corner case detection for autonomous driving", in *IEEE Intelligent Vehicles Symposium (IV)*, 2019, pp. 438–445.

[63] J. Breitenstein et al., "Systematization of corner cases for visual perception in automated driving", in *IEEE Intelligent Vehicles Symposium (IV)*, 2020, pp. 1257–1264.

[64] J. Breitenstein et al., "Corner cases for visual perception in automated driving: Some guidance on detection approaches", *arXiv preprint arXiv:2102.05897*, 2021.

[65] T. Ouyang et al., "Corner case data description and detection", in *IEEE/ACM 1st Workshop on AI Engineering-Software Engineering for AI (WAIN)*, 2021, pp. 19–26.

[66] T. Ouyang et al., "Improved surprise adequacy tools for corner case data description and detection", *Applied Sciences*, vol. 11, p. 6826, 2021.

[67] W. Wu et al., "Deep validation: Toward detecting real-world corner cases for deep neural networks", in *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, IEEE, 2019, pp. 125–137.

[68] F. Heidecker et al., "An application-driven conceptualization of corner cases for perception in highly automated driving", in *IEEE Intelligent Vehicles Symposium (IV)*, 2021, pp. 644–651.

[69] F. Heidecker, M. Bieshaar, and B. Sick, *Towards corner case identification in cyclists' trajectories*, 2019.

[70] A. Le Coz et al., "Leveraging generative models to characterize the failure conditions of image classifiers", in *The IJCAI-ECAI-22 Workshop on Artificial Intelligence Safety (AISafety 2022)*, 2022.

[71] A. Awadid et al., "A methodological framework for supporting the operational analysis of ML-based systems", in *Models and Methods for Systems Engineering*, Springer, 2025, pp. 129–141.

[72] A. Fakhouri et al., "Ml model coverage assessment by topological data analysis exploration", in *Proceedings of the AAAI Symposium Series*, vol. 4, 2024, pp. 32–39.

[73] E. Breck et al., "The ml test score: A rubric for ml production readiness and technical debt reduction", in *IEEE International Conference on Big Data (Big Data)*, 2017, pp. 1123–1132.

[74] J. Thomas, "Systems theoretic process-analysis STPA", *Availble online at: http://psas. scripts. mit. edu/home/wp-content/uploads/2016/01*, 2016.

[75] J. Berger, *STPA guide*. VTT Technical Research Centre of Finland, 2024.

[76] B. Braunschweig et al., "AITA: AI trustworthiness assessment: AAAI spring symposium 2023", *AI and Ethics*, vol. 4, pp. 1–3, 2024.

[77] K. Kapusta et al., "Protecting ownership rights of ml models using watermarking in the light of adversarial attacks", *AI and Ethics*, vol. 4 - 1, pp. 95–103, 2024.

[78] M. Lansari et al., "A Black-Box Watermarking Modulation for Object Detection Models", in *Proceedings of the AAAI Symposium Series*, vol. 4, 2024, pp. 60–67.

[79] J. Mattioli et al., "An overview of key trustworthiness attributes and KPIs for trusted ML-based systems engineering", *AI and Ethics*, vol. 4 - 1, pp. 15–25, 2024.

[80] A. Dakwat and E. Villani, "System safety assessment based on STPA and model checking", *Safety science*, vol. 109, pp. 130–143, 2018.

[81] A. Awadid and B. Robert, "On assessing ML model robustness: A methodological framework", in *Symposium on Scaling AI Assessments*, 2025.

[82] J. Mattioli et al., "Towards a holistic approach for AI trustworthiness assessment based upon aids for multi-criteria aggregation", in *SafeAI 2023-The AAAI's Workshop on Artificial Intelligence Safety*, vol. 3381, 2023.

[83] J. Mattioli et al., "A Brief Overview of Key Quality Metrics for Knowledge Graph Solution. Illustration on Digital NOTAMs", in *Proceedings of the AAAI Symposium Series*, vol. 7, 2025, pp. 206–213.

[84] S. Naveed et al., "An overview of the empirical evaluation of explainable AI (XAI): A comprehensive guideline for user-centered evaluation in xAI", *Applied Sciences*, vol. 14, p. 11 288, 2024.

[85] J. Mattioli et al., "Leveraging tropical algebra to assess trustworthy AI", in *Proceedings of the AAAI Fall Symposium Series*, vol. 4, 2024, pp. 81–88.

[86] C. Labreuche, "A general framework for explaining the results of a multi-attribute preference model", *Artificial Intelligence*, vol. 175, pp. 1410–1448, 2011.

[87] M. Ducoffe et al., "Lard–landing approach runway detection–dataset for vision based landing", *arXiv preprint arXiv:2304.09938*, 2023.