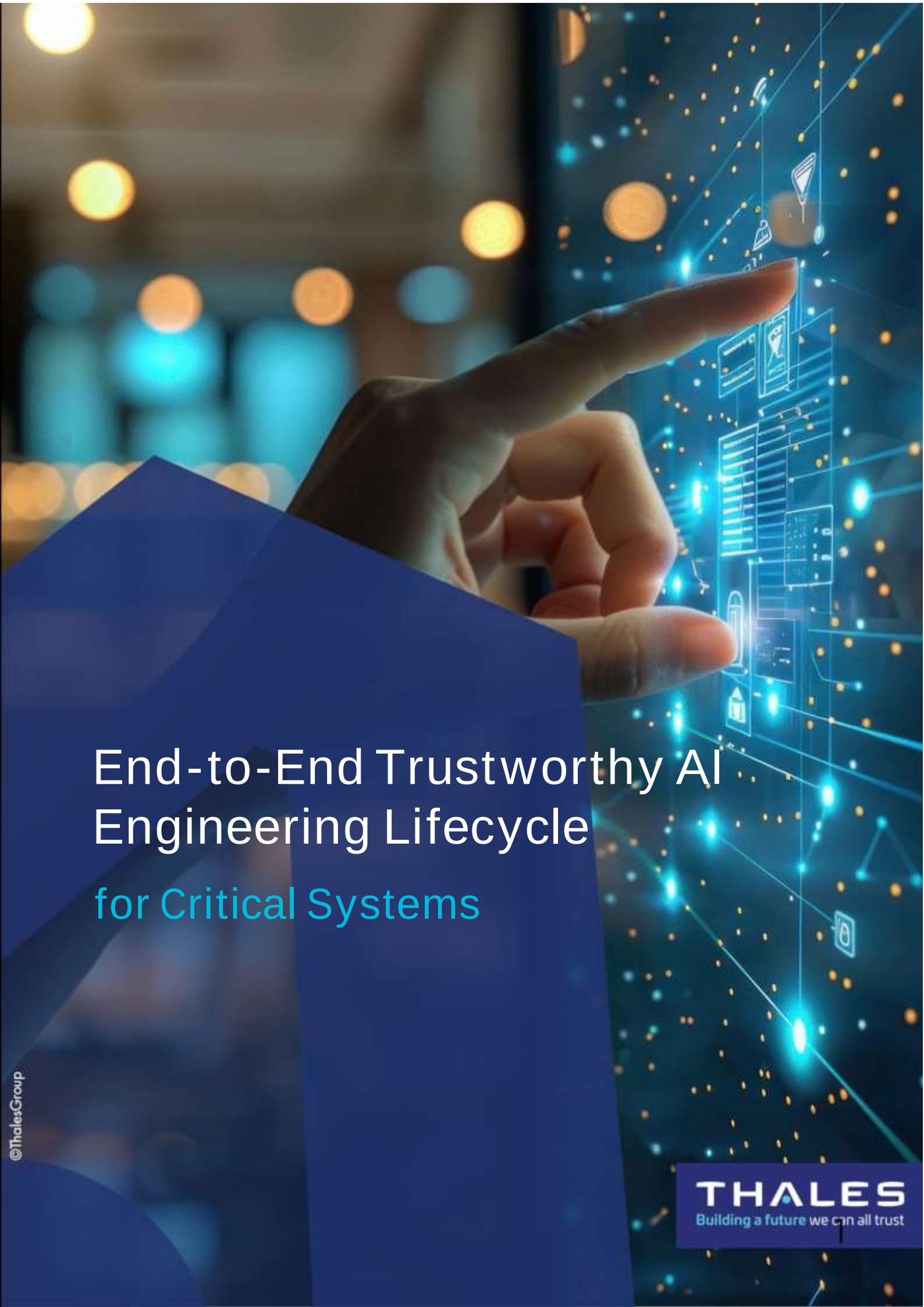

End-to-End Trustworthy AI Engineering Lifecycle

for Critical Systems

THALES
Building a future we can all trust

THALES
Building a future we can all trust

End-to-End Trustworthy AI Engineering Lifecycle for Critical Systems

Juliette MATTIOLI, Patricia BESSON, Fateh KAAKAI,
Michel BARRETEAU, Rémi BLANCHETTE, Fabien FLACHER

Summary

The End-to-End Trustworthy AI Engineering Lifecycle presented in this white paper underscores the necessity of a holistic and rigorous approach to the development, deployment, and maintenance of AI-powered systems in critical domains like aeronautics and space, defence, and security and identity. From the foundational principles of system engineering—encompassing safety, security, ethics, and system operational domain characterization—to the specialized considerations of AI algorithm engineering and implementation, this lifecycle provides a comprehensive framework for addressing the complexities and challenges inherent in AI technologies.

The integration of MLOps / AIOps and DevOps practices into the engineering lifecycle further reinforces the adaptability and reliability of AI systems, fostering seamless collaboration among data scientists, engineers, and operational teams, and enabling continuous integration, deployment, and monitoring of AI models. Similarly, the use of AI-powered tools in engineering activities enhances the efficiency of requirement development, testing processes, and code generation. However, the necessity of a standardized qualification framework for these tools is crucial to ensure that they contribute to, rather than compromise, the reliability, safety and security of critical systems.

Looking ahead, the challenges posed by AI's increasing complexity and its integration into mission-critical systems require sustained collaboration between industry, regulatory authorities, and academia. The discussion on new paradigms, such as runtime assurance, incremental development & qualification, and AI services highlights the evolving needs of AI systems to remain adaptive in dynamic environments while ensuring trustworthiness. These strategies are particularly critical in domains where operational constraints demand rapid updates, emphasizing the balance between safety, performance, and mission-critical objectives.

Through the end-to-end trustworthy AI engineering lifecycle, Thales reaffirms its commitment to developing AI-powered systems that are not only efficient, safe and secure, but also responsible, transparent and ethical. cortAIx, as Thales' AI accelerator, plays a central role in research activities to invent and advance new concepts, algorithms, assurance methodologies and engineering tools for the industrialization of AI products and services. By constantly pushing the boundaries of AI technology, Thales is setting new benchmarks of excellence in mission-critical systems applications, reshaping the future of AI in the most demanding and contested environments.

1. Introduction and Context

The rapid adoption of Artificial Intelligence (AI) is revolutionizing operational capabilities in critical domains such as aeronautics and space, defence, and security and identity [12]. However, this transformation comes with significant challenges to ensure that AI-powered systems are trustworthy, i.e. valid, explainable, robust and responsible[1]. Cutting-edge AI applications taking advantage of machine learning, symbolic AI, and hybrid AI (explored later in this paper) must deliver robust, evidence-based solutions for critical domains. These advancements demand a structured framework for substantiating trust in AI systems through rigorous engineering practices, an approach that is the focus of this white paper.

On the international level, the European Union Aviation Safety Agency (EASA) has outlined a comprehensive vision for trustworthy AI in its AI Roadmap and Concept Paper, which emphasize safety, transparency, and accountability. Complementing these efforts Thales is strongly involved in international standardization initiatives such as those led by EUROCAE WG-114[2] in collaboration with SAE G34, which provide guidelines for certifying AI in critical systems. ISO and CENELEC are also contributing to the global standardization landscape with their cross-sectors standards for ethical and reliable AI applications. On the French national level, Thales is one of the founding members with IRT SystemX and other industrial partners of the French project Confiance.ai[3], a pillar of the Grand Défi ''AI for industry'' initiative, which is pioneering methodologies for the development of trustworthy AI systems across sectors.

This white paper introduces the End-to-End Trustworthy AI Engineering Lifecycle, a holistic approach to designing, developing, and deploying AI-based systems in critical environments. By integrating the principles outlined in international and national initiatives with internal advanced engineering practices, this lifecycle ensures that AI systems not only perform their intended functions, and only their intended functions, with the desired level of performance but also makes AI-powered solutions transparent, responsible and ethical. The paper further delves into how these AI engineering frameworks can be operationalized within the existing standardization ecosystem, providing actionable insights for Customers and stakeholders including engineers, researchers, and policymakers.

This algorithm engineering layer acts as a bridge, translating complex system requirements into sophisticated AI models while ensuring compatibility with existing system engineering processes. It introduces tools and methodologies tailored to the iterative nature of AI development, enabling frequent updates and refinements as new data or insights become available. Additionally, this algorithm-engineering layer aims to standardize interfaces between AI algorithms and software / hardware layers, ensuring seamless integration and reducing the risk of errors during implementation and deployment [1].

The introduction of this new layer has significant implications for existing engineering workflows, which are often highly standardized. By aligning with established standards, such as those governing system and software engineering, the Algorithm Engineering Layer ensures that AI development remains consistent and traceable. Furthermore, it fosters collaboration among multidisciplinary teams, enabling system engineers, data scientists, and software developers to work synergistically toward common goals. This holistic approach enhances the reliability and trustworthiness of AI systems, ensuring they meet the stringent requirements of critical applications.

[1] According to Thales TRuE AI definition for trustworthy AI
[2] Thales is co-charing this EUROCAE working group with Airbus
[3] Further details about this project can be found at Confiance.ai

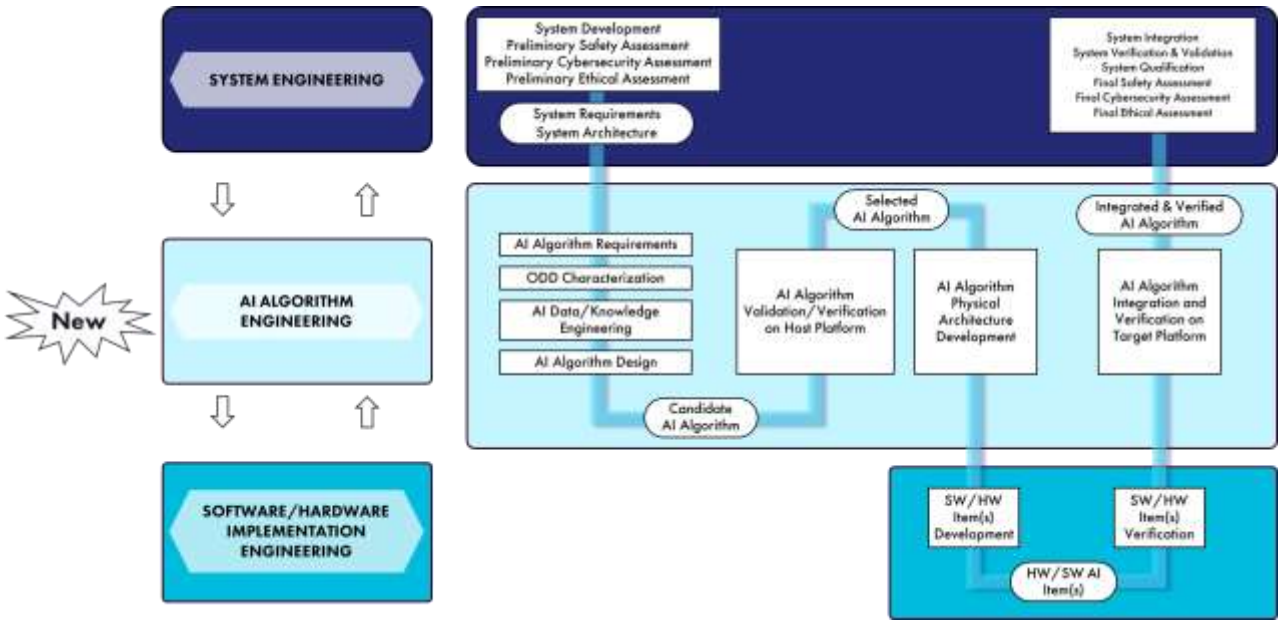

Figure 1 – Specific AI Algorithm Engineering Layer

2. End-to-End Trustworthy AI Engineering Lifecycle

3.1 SYSTEM ENGINEERING

The development of AI-based systems for critical applications [12], whatever the AI technology that is used (connectionist and statistics AI, symbolic AI or hybrid AI), necessitates rigorous System Engineering practices where reliability, availability, maintainability, safety, cybersecurity, usability and ethics are key considerations. To reach that objective, the ''Thales TrUE AI'' approach was set up in 2019, comprising **Tr**ansparent AI (where the relevant data used to arrive at a conclusion are presented to the users), **U**nderstandable AI (that can explain and justify the results) and finally an **E**thical AI, (following objective standards protocols, laws, e.g., the European AI Act, and human rights, where validity, explainability, security, and responsibility are prerequisites for moving towards the qualification, approval, and even certification of a critical AI-based system). To achieve these goals, system engineering for AI-based applications combines traditional engineering practices with AI-specific considerations.

System Operating Domain

This is a crucial step, which shakes up the habits of AI researchers or engineers: it includes the rigorous description at system level of all foreseeable operating conditions, called the system-operating environment, to enable a suitable data collection and knowledge representation. AI algorithm reliability directly depends on the correctness and completeness of the system operating environment description, in particular events that necessitate specific awareness such as rare events, combinations or sequences of operating conditions with potential safety implications. Indeed, the validity of a system can only be established for the intended use, i.e. the operating domain.

Safety

Ensuring safety (and beyond reliability, availability, maintainability) means that AI systems must perform and must keep performing over time as intended under all foreseeable operational conditions, including edge / corner cases and failure scenarios, to avoid unintended consequences that could jeopardize human lives or mission-critical functions. Hazard analysis and risk assessment are tailored to AI's unique characteristics, such as critical errors or biases in training data or knowledge representation and the capability of an AI model to generalize to unseen data within the operational design domain [2]. The performance requirements allocated to the AI algorithm may be driven by safety objectives to limit its worst credible approximation error to a given acceptable threshold.

Security

Cybersecurity, a cornerstone of trustworthiness, requires robust measures to protect AI systems against adversarial

attacks, data breaches, and unauthorized manipulation. These systems must incorporate advanced threat detection and mitigation strategies, as well as resilience mechanisms that enable them to operate securely in dynamic, hostile environments. Cybersecurity measures should be embedded not only at the system level but also within the underlying data pipelines, ensuring that data integrity and privacy are maintained throughout the AI lifecycle. Note also that boundaries between security and safety is not a tight border in an AI context: changes in the model inputs leading to wrong outputs can come from intentional malicious actions, or from unexpected natural conditions.

Figure 2 – Thales TrUE AI [Transparent, Understandable, and Ethical AI]

3.2 AI ALGORITHM ENGINEERING

Machine Learning

For connectionist and statistics AI, especially for Machine Learning (ML), the AI Algorithm is composed of different ML building blocks which are one or many ML Model(s) and the associated data processing. These ML building blocks are then integrated into the system to which it belongs.

The development of ML-based systems can be visualized as a W-shaped lifecycle [2]. The W-Shape can be divided into two parts: the 1st V includes the ML engineering processes performed on the Host Platform (e.g. on a private computing cluster, on a secured partition on the cloud), and the second V includes the implementation engineering processes performed on the Target Platform (e.g., specific hardware embedded in a ground or aerial vehicle).

In the first V, the ML engineering lifecycle [1][2][4] begins with defining AI / ML Algorithm requirements refined from system specification. This ML specification step is completed by the characterization of the Operational Design Domain (ODD). The ODD description is developed, on the one hand, as a refinement of the portion of the system operational domain allocated to the AI / ML Algorithm (top-down approach), and on the other hand, as a data-centric analysis (bottom-up approach). The ODD is a means to reconcile data and functional intent, that is to align the data used for the training and the resulting ML Model(s) with the intended operational usage, capturing a wide spectrum of operating conditions it may encounter [5].

Data engineering plays a foundational role, involving the identification, collection, preprocessing, and feature extraction of extensive datasets essential for the design of ML Models and their verification. This phase often incorporates advanced techniques such as data augmentation, synthetic data generation, and anomaly detection to enhance the dataset's representativeness, completeness and relevance (minimization of the simulation-to-reality gap). Rigorous quality control processes driven by data quality requirements (DQRs) ensure the quality of data inputs (e.g. biais mitigation, privacy/sovereignty...), enabling accurate and consistent training outcomes. During ML Model Design, engineers focus on selecting the most appropriate learning algorithms, crafting model architectures, and refining them through iterative cycles of training and evaluation. Furthermore, optimization strategies such as pruning, quantization, and model distillation are applied

to balance computational efficiency with performance. Validation and Verification (V&V) activities are driven by key trustworthiness properties such as ML Model stability, generalization, robustness, and reproducibility. These ''good properties'' are specified into low-level ML requirements with the definition of satisfaction criteria, metrics and confidence scores to handle uncertainty related to data, model approximation, and statistical assumptions. Validation activities are mainly conducted to ensure the correctness and completeness of the ML requirements through review, analysis and traceability with the upper layer of requirements. Verification activities include extensive simulation, robustness testing of edge / corner cases, scenario-based testing, ML model explainability analysis, and ODD coverage analysis. The 1st V ends with a selected AI / ML Algorithm that 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 (e.g., FPGA) items in the 2nd V (that will be further detailed in the section 3.3).

Symbolic AI Engineering

For Symbolic AI, it is essential to classify its techniques into two distinct groups, each with different implications for development, qualification, and certification. The first group comprises straightforward symbolic AI techniques already integrated in existing systems that have been qualified and certified using existing standards. Examples include rule-based systems for decision-making, deterministic logic frameworks, and simple expert systems such as those used in predictive maintenance and health monitoring. These systems operate within well-defined boundaries and rely on straightforward, interpretable rules, making them amenable to conventional verification and validation methods. The second group encompasses more complex symbolic AI techniques designed for implementing intricate system functions. These include advanced reasoning engines, multi-layered ontologies, and dynamic logic programming used for adaptive planning, strategic decision-making, or large-scale knowledge integration. These systems often exhibit non-linear behavior and require a higher degree of abstraction and interaction, posing challenges for traditional qualification and certification frameworks. Consequently, this group demands a new paradigm, like the new AI Algorithm engineering layer of the W shape lifecycle, for qualification and certification to address their complexity and ensure safety, reliability, and compliance in critical system contexts.

For this second group of Symbolic AI technologies, the engineering process is rooted in a knowledge-driven paradigm, beginning with the translation of high-level system requirements into structured AI algorithm requirements. This involves crafting logical architectures and detailed scenario descriptions that thoroughly capture the problem domain and operational nuances. Engineers focus on selecting the most appropriate knowledge representation and reasoning algorithm during the Model Design phase. Thus, during the AI Data / Knowledge Management phase, efforts are concentrated on systematically organizing domain knowledge into structured knowledge bases, employing sophisticated schema design, ontologies, and hierarchical rule-based frameworks. These structures ensure modularity, accessibility, and scalability for reasoning tasks. The symbolic AI model design phase involves developing detailed inference rules, constructing robust logic structures, and integrating advanced knowledge representation mechanisms tailored to the application's needs. Tools such as symbolic solvers, decision trees, and optimization algorithms can be deployed to refine these models for accuracy and efficiency. Validation and verification for Symbolic AI leverage scenario-based simulations of tractable complexity combined with formal methods and model checking to ensure logical consistency, correctness, and alignment with predefined requirements.

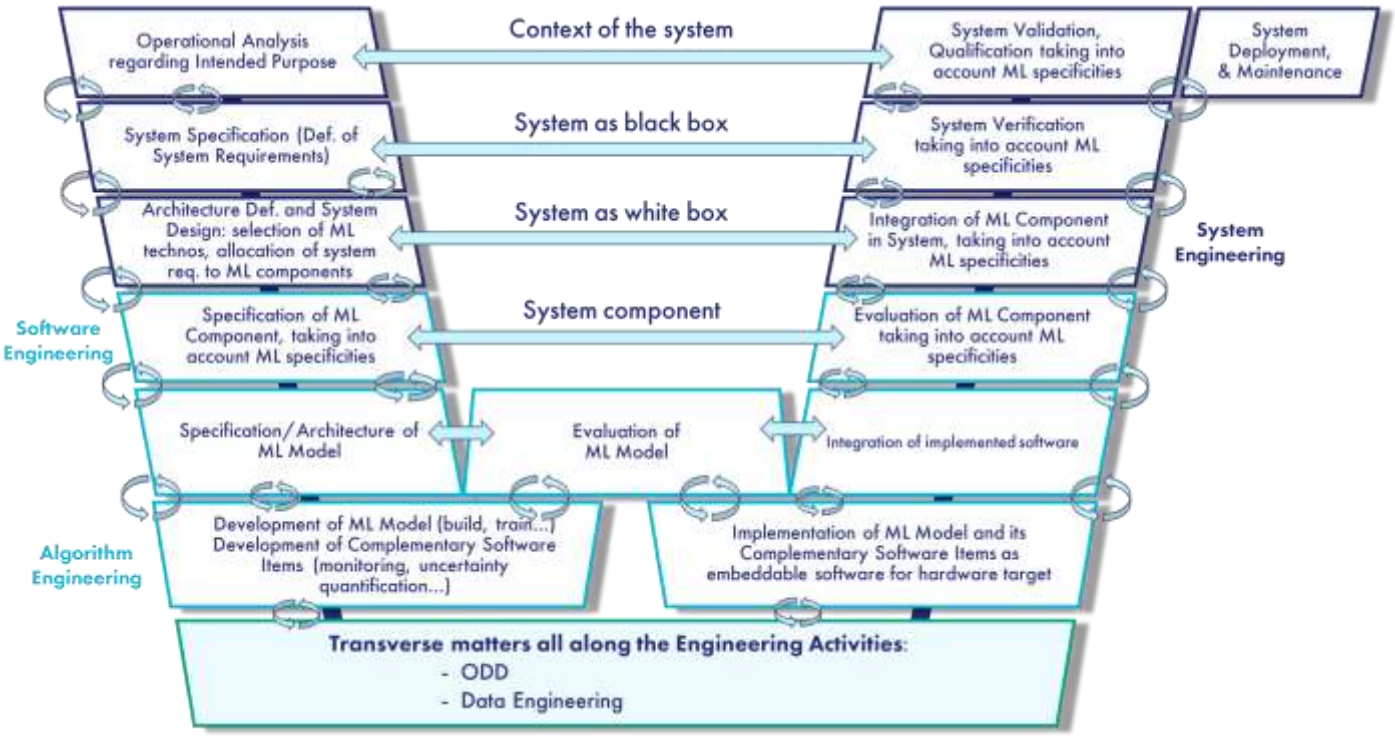

Figure 3 - AI W-Shape lifecycle for Machine Learning (source: https://bok.confiance.ai)

Hybrid AI Engineering

Hybrid AI combines Machine Learning (ML) and Symbolic AI to create robust, efficient, and explainable systems by leveraging their complementary strengths. ML excels in processing unstructured data, recognizing patterns, and adapting through learning, while Symbolic AI offers strengths in structured knowledge representation, reasoning, and interpretability. In other words, the efforts deployed in the Knowledge Management phase for Symbolic AI is just as relevant to enhance the overall ability of a ML-based model. To harness these capabilities, a unified hybrid AI engineering lifecycle integrates methodologies from both paradigms into a cohesive framework.

The lifecycle begins with requirement development and operational domain characterization, balancing data-driven and rule-based components. ML relies on large datasets to define behavior within the Operational Design Domain (ODD), whereas Symbolic AI uses logical structures and inference rules for reasoning. Hybrid systems require integrating knowledge bases with datasets, enabling interaction between symbolic reasoning and ML learning processes.

In the design phase, ML model architectures align with symbolic inference mechanisms. Engineers define integration points where knowledge bases (e.g., physical equations or geometric properties) or symbolic reasoning (e.g., ontologies) inform ML or vice versa. Hybrid workflows are optimized by refining ML models with symbolic feedback or using ML to supplement symbolic knowledge bases. Validation and verification may combine ML simulation and dataset testing with formal methods to ensure symbolic AI consistency, requiring innovative strategies to test hybrid interactions across varied scenarios.

3.3 SOFTWARE/HARDWARE IMPLEMENTATION ENGINEERING

The implementation of AI algorithms in critical systems must adhere to existing (and incoming) standards[4], with specific considerations for AI technologies.

Machine Learning

For ML, the AI algorithm implementation phase involves developing a physical architecture (software / hardware items) of the AI algorithm. According to its complexity, the engineers may need to break down the AI algorithm into

[4] Existing standards such as ED-12C [9] and its supplements or ED-80 [10] in the aeronautical domain; EUROCAE WG114 / SAE G34 Artificial Intelligence In Aviation AS6983 Process Standard For Development And Certification / Approval Of Aeronautical Safety Related Products Implementing AI

several pieces, called items, to meet implementation constraints imposed by the target hardware platforms such as GPUs or embedded platforms using accelerators like the Thales Neural Processor (TNP). Automated test pipelines are used to verify the implementation of the optimized AI algorithm in one or more software and / or complex electronic hardware (CEH). The approach is to run the test dataset(s) used during the design stage (1st V) complemented by additional specific test cases to evaluate performance properties such as Worst-Case Execution Time (WCET), latency, memory footprint and power consumption on the target platform(s). This additional verification phase carried out at the end of the 2nd V (see Figure 3) ensures that no implementation or integration errors (regressions), nor any incompatibility with the target platform(s) degrade the desired performance of the AI Algorithm.

Symbolic AI Engineering

For Symbolic AI, implementation transitions logical architectures into physical ones, ensuring compatibility between computational resources for symbolic AI reasoning and broader system architecture and resources. Verification processes must confirm that symbolic AI items meet their requirements including performance, robustness, safety, and security. These processes include verifying inference accuracy, ensuring consistency within and between implemented knowledge representations, and testing scalability under dynamic operational conditions. By integrating these principles, symbolic AI systems can deliver reliable reasoning capabilities even in highly complex or evolving environments.

Hybrid AI Engineering

In Hybrid AI systems, implementation integrates ML models and symbolic reasoning components, often requiring specialized hardware for logic processing alongside ML accelerators. This dual approach necessitates careful design of interfaces to ensure seamless integration and compatibility between these two complementary technologies. Modular programming practices enable these systems to operate effectively across diverse scenarios. Verification activities include the ones mentioned in the previous sections of Algorithm Engineering dedicated to ML and symbolic AI completed by interface testing, scenario-based simulations, and stress testing, ensuring hybrid AI systems achieve the combined objectives of performance, robustness, scalability, and compliance with safety, security and regulatory standards.

3.4 GOING FURTHER: CROSS-DISCIPLINARY CHALLENGES

This section briefly describes some of the major interdisciplinary challenges within computer sciences and software / system engineering, which require particular attention in the context of critical systems.

Trustworthy AI Assessment

Assessing trustworthiness in AI is key for the design and operation of critical systems to improve the overall life cycle of such systems (from requirements to maintenance through design, deployment and operation). To make the system more trustworthy, we need to assess it and set up strong and rigorous processes. The first issue is to measure how trustworthy AI-based systems are. This involves using existing metrics and assigning scores to different attributes. Usually, it is determined by the quantification of elementary scores (e.g., for reliability: Fleiss Kappa score, goodness-of-fit tests, or for accuracy: precision, recall, F-score...). From a strict metrological point of view, trustworthiness may not be measured, as it is not a physical property that can be compared to a reference quantity of the same kind. Trustworthiness does not have units. The Confiance.ai program [13] describes different attributes that contribute to the concept of trustworthiness, looking closely at each attribute to identify the key performance indicators (KPI), evaluation methods or checkpoints, and establishing an aggregation methodology for these attributes. Another issue is how the system will be used. Context is important when it comes to trustworthiness. This is shaped by a multitude of factors. For example, a critical AI-based system in aeronautics may have different trustworthiness requirements than in automotive or health care. Moreover, some attributes may be quantitative, typically comprising numerical values derived from measurements or providing a comprehensive statistical overview of a phenomenon. Thus, it is not straightforward to select the relevant attributes for assessing AI trustworthiness, given that the choice depends on usability and on the context of application. This context is modeled according to several elements, including the operational design domain, the intended domain of use, the nature and roles of the stakeholders, and so on.

Runtime Assurance to Complement Design-time Assurance

In system engineering, assurance refers to systematic actions to provide confidence that a system meets specified requirements. This structured process minimizes development errors contributing to potential failure

conditions, such as safety or security issues, through rigorous assessment carried out at design-time known as the development assurance process [8]. After deployment, runtime assurance ensures a system behaves correctly during execution, and it involves real-time monitoring of the system's behavior and predefined corrective actions for deviations from the desired system's behavior. This is especially crucial for critical systems, where failures could have severe consequences.

Key elements of runtime assurance for AI-powered solutions include monitoring operational behavior to detect anomalies[5] and assertion checking to verify system conditions during execution. Recovery strategies provide automated responses to errors, from simple alerts to switching to backups or reducing functionality for safety purposes (fail-safe or fail-stop paradigm according to the domain). For systems operating within distributed or decentralized environments, such as drone swarms or networks of autonomous vehicles, runtime assurance must extend beyond individual components to encompass the entire collective. This necessitates mechanisms to coordinate runtime assurance mechanisms across multiple entities, ensuring that safety, security, and performance are maintained at the swarm or network level. Such approaches involve collaborative monitoring, synchronized decision-making, and distributed fault tolerance to address the unique challenges posed by these interconnected systems. Another important consideration is the design of runtime assurance mechanisms that are sensitive to the resource constraints of the system, such as computational power, memory, energy, size and weight. This is crucial for ensuring that assurance processes do not overly burden the system's ability to perform its primary critical functions.

Runtime assurance complements design-time verification by addressing unforeseen issues during real-world operation. While design-time methods prevent development errors such as incorrect requirements or coding bugs, runtime assurance adds a dynamic safeguard for handling unexpected scenarios. With the rise of AI and autonomy, the reliance on a posteriori runtime assurance plays, already today, and will continue to play in the future, a larger role in compensating for the limitations of a priori development assurance activities. As AI and autonomous capabilities increase, so does the importance of runtime assurance, as illustrated in the next figure.

From DevOps and MLOps to AIOps

To support that shift in assurance, the engineering practices that have historically stopped at design time will need to extend in the runtime, as systems evolve during their lifespan. The integration of DevOps (Development Operations) and MLOps (Machine Learning Operations) practices is critical to support AI algorithm engineering industrialization in that evolving context. DevOps provides a framework for automating software development and deployment workflows, ensuring faster iterations and consistent quality. MLOps extends the principles of DevOps by addressing the unique challenges of ML technologies, such as managing data pipelines, orchestrating model training, and monitoring model performance post-deployment. Together, these practices streamline the lifecycle of AI algorithms by enabling continuous integration and continuous deployment (CI/CD), version control for models and data, and automated testing to detect potential issues early. MLOps ensure that AI model's performance remains in optimal condition by integrating retraining workflows triggered by new data or changing operational conditions, reducing the risk of model drift. Meanwhile, DevOps complements this by managing the broader software infrastructure, ensuring that updates to ML models align seamlessly with other system elements. The synergy between DevOps and MLOps enhances collaboration between data scientists, ML engineers, software engineers and operations teams, driving efficiency and reducing time-to-market for AI solutions. Thales works on extending this holistic approach to symbolic AI and hybrid AI to build a general framework, called AIOps[6], which is essential for building and maintaining complex systems involving different AI technologies, particularly in critical applications where rapid adaptation and robust performance are non-negotiable. To deliver state of the art robust AI systems, this multidisciplinary process should be tooled, in design and deployed environments, in order to streamline the collaboration between the various teams and disciplines.

[5] Anomaly stands for outside of the ODD

[6] MLOps stands for Machine Learning Operations, Even though AIOps is often referred to ''artificial intelligence for IT operations'', in this document, AIOps is the extension and adaptation of DevOps principles to the AI ecosystem, including statistical and connexionist AI, symbolic AI, hybrid AI, generative AI and agentic AI.

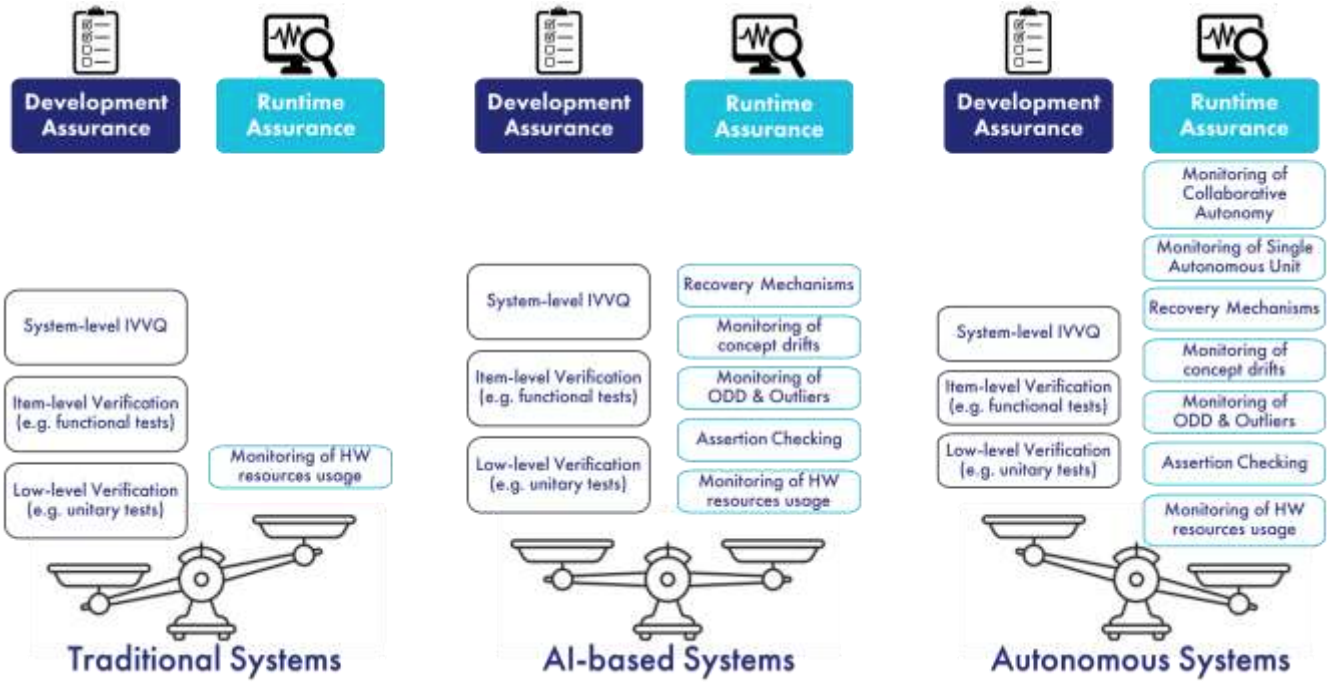

Figure 4 – Development Assurance versus Runtime Assurance

Incremental Development and Qualification

By adopting incremental development and qualification, both defence and civilian domains can address frequent operational changes and remain adaptive and safe, maintaining their performance in critical applications.

Defence applications, like automatic target detection and recognition (ATDR) function implementation, demand frequent and rapid updates to AI models to counter evolving concealment maneuvers from the enemy. For instance, overnight, enemies may camouflage tanks with vegetation, impacting AI detection capabilities and increasing false negatives. Damage caused by military attacks also continuously changes the battlefield environment (e.g., craters, debris, dust that changes the color of infrastructure and vegetation). Rapid incremental development and qualification processes ensure that AI models can adapt to maintain operational effectiveness.

In civilian applications, such as urban mobility and air traffic management, AI models must adapt to dynamic conditions like shifting weather or new obstacles and must align with the temporal validity of training data. Delivery drones, for example, may need to navigate unexpected construction zones. In the domain of air traffic management, systems must integrate real-time updates to address emergent flight delays or weather changes, and therefore maintain high performance without operational disruptions, balancing safety and efficiency.

However, updating AI models rapidly through effective incremental changes and risk management procedures is infeasible by following, for each increment, the entire development lifecycle (e.g., W-shape lifecycle) due to operational time constraints and cost. In this respect, industrials must make it a priority to conduct research into new methodologies in collaboration with the authorities responsible for system approval and certification. These methodologies aim to formalize the trade-offs necessary between maintaining a sufficient level of safety and security and achieving strategic mission objectives, and to assess the inherent risk generated by these trade-offs for reporting to the appropriate authorities, such as military command.

From AI products to AI Services

The development of AI services to complement or replace AI products is a strategic industrial necessity where frequent and rapid updates to AI models are required to maintain high performance and safety levels. AI Algorithms such as ML models are inherently dynamic systems that depend heavily on the quality, volume, and characteristics of the data they are trained on. Over time, factors such as shifts in user behavior, or operational variability can render static models ineffective, a phenomenon often referred to as concept drift. AI products, which are typically deployed as fixed solutions, lack the flexibility to accommodate these changes

efficiently, resulting in diminished performance and relevance over time.

In contrast, AI services adopt a fluid, iterative approach that allows continuous monitoring, retraining, and redeployment of models as needed. This paradigm recognizes that AI-powered systems are not "set-and-forget" solutions but require an infrastructure capable of incremental development and qualification. By leveraging modular architecture and cloud-based deployment pipelines, including sovereign environment for defence applications, AI services enable seamless updates without disrupting end-user operations. This ensures that the system remains robust and responsive to changing inputs, whether due to external factors like data drift or internal factors like shifts in operational priorities.

Additionally, the AI service business paradigm aligns with the increasing demand for scalability and interoperability in AI applications. Modern AI-based systems often operate within ecosystems where integration with other services and platforms is critical. The flexibility of AI services facilitates easier adaptation to new requirements, such as compliance with updated regulations or the incorporation of new data sources. Moreover, the service-oriented paradigm supports a feedback loop, wherein performance metrics and user feedback directly inform iterative improvements, fostering a system that evolves to meet emerging needs.

From an operational standpoint, the service model also supports cost efficiency and resource optimization. By deploying updates incrementally and focusing on specific modules rather than the entire system, organizations like Thales and its Key Customers can minimize downtime and reduce the risks associated with large-scale system upgrades. This approach is particularly advantageous in industries dealing with critical systems, where the cost of errors or delays is exceptionally high.

AI for Engineering

AI technologies are also increasingly recognized for their potential to enhance engineering activities across various phases of development lifecycle such as requirement engineering, testing, and coding. AI-powered tools can support requirements development and requirements validation, but it is not recommended to use AI in both activities. For requirements development, AI tools can analyze stakeholder inputs[7], and automatically generate structured requirements to cover the captured intent. Humans or any other means that is dissimilar with the AI tool (to avoid any common mode of failure) should validate these generated requirements. For AI-based validation of requirements developed by engineers, AI tools can cross-check these requirements against the stakeholders' inputs or previous project data to identify inconsistencies or completeness gaps. In the domain of testing, AI can automate the generation of test cases and test scenarios by leveraging advanced algorithms to simulate operational conditions, identify edge and corner cases, and ensure sufficient coverage of the ODD. Similarly, AI tools can accelerate source code development through automated code generation that translates high-level specifications into source code, provided that this automated source code is reviewed by software engineers. For the review of source code written by software engineers, AI tools equipped with natural language processing and static analysis capabilities can identify potential vulnerabilities, inefficiencies, or violations of coding standards, significantly improving the quality and safety of the developed software.

While these AI tooling advancements bring tremendous benefits, they also introduce the risk of propagating errors or biases inherent in the AI tools themselves. Without proper oversight and verification, these tools could inadvertently compromise the safety and reliability of the systems they are intended to support. To mitigate such risks, there is a critical need for a standardized approach to AI tool qualification, akin to the rigorous framework outlined in EUROCAE ED-215 [11] for software tools in critical system development. Such a standard would define criteria for evaluating AI tools, including their accuracy, robustness, and reliability, as well as methods for verifying their outputs. Additionally, it would establish requirements for traceability and transparency to ensure that any decisions or outputs generated by AI tools can be audited and understood by human engineers.

Adopting a standardized qualification framework for AI tools is essential to ensure that these technologies enhance engineering processes without introducing unacceptable risks in the development of critical applications.

[7] For example, at system level, stakeholder inputs may include Concept of Operations / Usage (ConOps / ConUse), regulatory standards, risk analyses addressing safety, security and human factors.

AI Governance

AI governance encompasses the rules, processes and practices that ensure the ethical, secure and compliant development and deployment of AI systems for its customers. This includes (but not limited to) defining the roles and responsibilities of the various players involved in the AI value chain and ensuring a rigorous data lifecycle from access to appropriate data / knowledge sources to archiving, retention and, if necessary, deletion and disposal of data. This AI governance also ensures compliance with ethical and regulatory requirements (e.g., in the European Union, the EU AI Act, GDPR). In specific areas such as defense, it ensures that processes and practices preserve data confidentiality as well as sovereignty requirements[8].

3. Thales Standpoint

Through critical AI-based systems [12], Thales is convinced that AI models will become more and more complex and be tasked with implementing increasingly sophisticated system functions (e.g., single pilot operations, collaborative target detection and recognition). Therefore, the need for a specific *algorithm-engineering layer* has emerged [2]. Positioned between traditional system engineering and software / hardware implementation, this new layer addresses the unique challenges posed by modern AI applications [3][6]. Unlike previous models, which were designed to execute small, well-defined technical tasks, today's AI systems must manage multifaceted functions in changing operating conditions that require higher levels of abstraction, flexibility, and integration. Thales cortAIx initiative is at the forefront of such end-to-end trustworthy AI engineering concerns.

4. References

[1] Confiance.ai, Towards the engineering of trustworthy AI applications for critical systems, Second Edition (2024)
[2] EASA. First Usable Guidance for Level 1 Machine Learning Applications. EASA Concept Paper Issue 01 (2021)
[3] EUROCAE WG114, Artificial Intelligence in Aviation Committee. AIR 6988, Artificial Intelligence in Aeronautical Systems: Statement of Concerns. SAE International (2021)
[4] Kaakai, F., Adibhatla, S., et al. Toward a machine learning development lifecycle for product certification and approval in aviation. SAE Int. J. Aerosp. 15 (2022)
[5] Kaakai, F., Adibhatla, S., Pai, G., Escorihuela, E. Data-Centric Operational Design Domain Characterization for Machine Learning-Based Aeronautical Products. SAFECOMP 2023. Springer, Lecture Notes in Computer Science, vol 14181 (2023)
[6] Delseny, H. et al. White Paper Machine Learning in Certified Systems. ArXiv abs / 2103.10529 (2021)
[7] Kaakai, F. and Raffi, PM.. Towards Multi-timescale Online Monitoring of AI Models. SafeAI@AAAI (2023)
[8] EUROCAE ED-79B, Guidelines for Development of Civil Aircraft and Systems. SAE International (2023)
[9] EUROCAE ED-12C, Software considerations in airborne systems and equipment certification (2012)
[10] EUROCAE ED-80, Design Assurance Guidance for Airborne Electronic Hardware (2000)
[11] EUROCAE ED-215, Software Tool Qualification Considerations (2012)
[12] Mattioli, J., Meyer, C. (2024) Artificial Intelligence for critical system, Thales Position Paper.
[13] Mattioli, J., Sohier, H., Delaborde, A et al. (2024). An overview of key trustworthiness attributes and KPIs for trusted ML-based systems engineering. AI and Ethics, 4(1), 15-25.

[8] e.g. such as in UK the UK MoD's JSP 936

THALES

Building a future we can all trust

4, rue de la Verrerie 92190
Meudon FRANCE

Tél. + 33(0)1 57 77 80 00

www.thalesgroup.com