AI for Future Skies: On-going standardization activities to build the next certification/approval framework for airborne and ground aeronautical products

Christophe Gabreau
Airbus
christophe.gabreau@airbus.com

Béatrice Pesquet-Popescu
Fateh Kaakai
Baptiste Lefevre
Thales
beatrice.pesquet@thalesgroup.com
fateh.kaakai.e@thalesdigital.io
baptiste.lefevre@fr.thalesgroup.com

Abstract

This position paper will describe the stakes of the new standard developed by EUROCAE and SAE on AI certification by detailing the main challenges, drawing the interfaces with existing standards, and proposing a new machine learning (ML) development lifecycle to support the future certification/approval objectives that will enable the use of ML techniques in the development of safety-critical applications for both airborne and ground aeronautical products.

1 Introduction

Artificial Intelligence (AI) is poised to transform the aerospace industry, impacting all areas in which computing and aerospace intersect. AI will embed into and transform the digital systems used to design, manufacture, operate, communicate and maintain aerial vehicles, and when leveraged successfully, it will dramatically change how aerospace companies operate, disrupting businesses while radically accelerating the pace of change. Specifically, Machine Learning (ML) technologies have the potential to revolutionize the development paradigms of aeronautical systems including the ones that are safety-critical. Current industrial guidance has a strong focus on bespoke technologies in aeronautical applications thus are not appropriate to support this paradigm change. Industrial guidance on AI development coming from other sectors (automotive in particular) are of great value, but are not directly applicable to the aeronautical industry, that has a long history of aviation-specific development standards.

Anticipating a growing commercial pressure for AI solutions within the aerospace industry, there was an urgent call for regulation to make emerge new norms around acceptable usage that fit in the existing aviation regulatory landscape.

1.1 Presentation of G-34/WG-114 working group

The EUROCAE WG-114 (joint with SAE G-34) was created to help guiding safe and successful adoption of AI technologies in Aeronautical Systems by developing industry consensus standards. The working group [1] is evaluating key applications for AI usage within aeronautical systems, with a focus on AI embedded into aerial vehicle and deployed on ground equipment, in order to produce standards for the development of safe systems compliant with regulation requirements.

The joint group has quickly grown and is worldwide with more than 500 engineers nowadays. It brings together the industry and the regulators, so that the resulting consensus standard could be recognized by certification authorities and used by industry. It draws its expertise from a large variety of industry fields, such as large aircraft manufacturers, but also Unmanned Aircraft Systems/Urban Air Mobility/ electric Vertical Take-Off and Landing manufacturers, engine manufacturers, airborne and ground equipment manufacturers, regulators, air navigation service providers and many more. There are big names such as Airbus, Boeing, EASA, FAA, NASA, DOD, Eurocontrol, Thales, Dassault, Safran, Nvidia, Intel… Large companies work alongside smaller companies and institutions. The working group organization has paid attention to correctly balance the domains representation with airborne and ground co-chairs at leadership and executive level. The standard will be preceded by some informative material and will include a description of use cases representative of the industrial needs (some of them will be used in the standard development to mature the guidance). By the end of 2020, the working groups established a "Statement of Concerns" in order to align all the aeronautical industries on the

[1] Acknowledgments: The work presented in this paper is the result of a collaborative work done by the EUROCAE WG-114 / SAE G-34 standardization working group. Special acknowledgement to the Subgroup 3 (SG3) core team that has actively contributed to the definition of the machine learning development lifecycle (sorted by alphabetical order of the last name): Shreeder ADIBHATLA (Rockdale Systems), Ramesh ANAPATHUR (Boeing), Elgiz BASKAYA (ENAC), Emanuele BEZZEC-CHI (Leonardo), Barclay BROWN (Raytheon), Konstantin DMITRIEV (TUM & MathWorks), Giacomo GENTILE (Raytheon), Stephane GRIHON (Airbus), Fabricio JOSE PONTES (Embraer), Christophe TRAVERS (Dassault-Aviation).

same concerns and raise the main challenges that prevent the use of AI and specifically the use of Machine Learning into safety critical applications nowadays.

1.2 Interfaces with existing standards

ML will be introduced to augment system functionalities. From a development process perspective, ML function will be part of existing systems and will have to be implemented in dedicated ML-based items. With analogy to the existing certification framework for model-based product, the framework for this new data-driven development paradigm will introduce a new layer in between the system level ([ED-79A/ARP4754A, 2011] for airborne, [2017/373] for ground) and the item level ([ED-12C/DO-178C, 2011], [ED-80/DO-254, 2000] for airborne, [ED-153, 2009], [ED-109A/DO-278, 2011] for ground). Clear interfaces will have to be defined in order to reuse existing development standards as described in Fig.1.

2 Certification/Approval Challenges

Among the important challenges the standard faces, in the first place there is the AI trustworthiness, for which the building blocks, as defined in the European Aviation Safety Agency (EASA) guidance [Cluzeau et *al.*, 2015], are the Learning Assurance, the AI Explainability and the AI Safety Risk Mitigation. They have been described in [ER-022/AIR6988, 2021] and can be further detailed in specific challenges, as follows:

2.1 Specification and Validation Challenges

Since their recent flourishing, the ML algorithms have been applied to all kind of tasks, but in particular to complex ones, where the previous approaches failed to attain the desired performances. These tasks are in themselves difficult to specify, due to their intrinsic complexity and ambiguity. The specification of a system involving ML components may be in itself an ill-posed problem, especially when using data-driven models. In this case the ML model is implicit, derived after the learning stage. The difficulty is therefore related to the black box nature of ML, which makes it impossible to apply the classical V-cycle, where the requirements can be traced down to the software code lines. Here, the code lines are very scarce, while the resulting model can be huge, and none of the low-level code line directly reflects specific requirements. This specification issue is intrinsic to *data-based* ML development, as opposed to the classical *model-based* development.

Moreover, the requirements validation also includes data requirements. Data requirements are difficult to specify completely, and therefore also to validate completely.

Another aspect of the ML black box model is its probabilistic nature, which will require to quantify, for the purpose of safety assessment, the uncertainties: the statistical model uncertainties, the dataset uncertainties, uncertainties related to the training process, and uncertainties induced by the implementation process. There is a limited theoretical capacity right now to prove all the properties that we would like about the robustness, stability and predictability of the ML models, but we can expect this aspect will be solved by the academic community, and some progress has been made recently [Katz *et al.*, 2017], [Combettes *et al.*, 2020], [Lattore *et al.*, 2020], [Gupta *et al.*, 2021]. In particular, a challenge is the testability of the data-based ML algorithms, which is related to the previous issues, and amounts to design adequate test datasets to prove robustness of the neural networks and ensure all the corner cases are covered.

Trusting a ML model (or building the three pillars of AI trustworthiness) involves « opening the black box» to a degree commensurate with its intended use, i.e., to the degree of safety (Assurance Level / Development Assurance Level / Software Assurance Level) required by the respective system or component. This will require new ways of specifying and validating data and model requirements, in order to build the

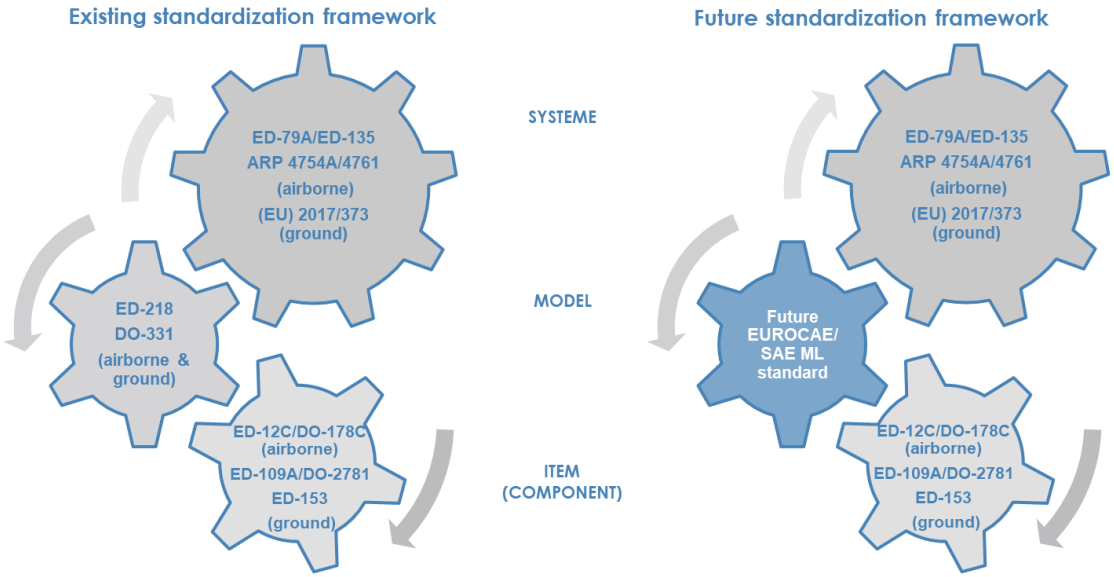

Figure 1 Standardization framework

ML model on a sound baseline of requirements, as described in section 3.

2.2 Data Challenge: Representativeness

The second main challenge is related to data. While a lot of data quality attributes have been identified as being relevant (accuracy, integrity, traceability, timeliness, accessibility …), the most important challenge derives from the need to correctly specify the problem, so it amounts to the data representativeness, its relevance to the problem, and its completeness for all situations that may be encountered [Rhie et *al*., 2017]. As the data itself can drive the ML function, quality, size, and composition of the datasets highly influence the behaviour of the system. Incorrect, incomplete or non-representative data, poor quality data or irrelevant features can significantly decrease the performance of the ML item even before training. If the performance of the system varies in different in specific circumstances (depending on the composition of the training dataset), this negatively affects the integrity of the system.

Ensuring representativeness of the training and testing data sets is far from being an obvious task, and the more we advance in the field of AI, the more we discover examples that show the potential bias into any collection of data, and also the variance that may come after inappropriate training with underfitted or overfitted datasets or simply insufficient specification of the problem and datasets.

We also witness a change in paradigm where the datasets may contain part of the specification while there is a conceptual need to have also, at system level, specifications independent from the dataset. Otherwise we would be trapped into a logic pitfall where we could not say that a bad dataset which became specification is bad, because it is the specification. Otherwise we would be unable to validate the dataset against the system needs.

2.3 Robustness and Verification Challenges

A very important challenge, which derives from the previous ones, but has a dominant importance in safety analysis, is related to the robustness and therefore to the verification of the ML models. The first question we need to assess is how variable is a model to the underlying training dataset. How much its characteristics vary depending on the way it was trained, the process in itself and the training dataset? And how robust it will be over time, once in production, to variations into its inputs?

This is related to a well-known robustness problem, first highlighted by the concept of « adversarial » attack, but that can be unforeseen as well. If a ML model is not robust, small variations of the inputs can lead to unexpected behavior of the model output [Gupta *et al*., 2021b], and [Latorre et al., 2020]. The role of the standard will be to give guidance on how to detect adversarial inputs and unintended outputs from the ML model. For this, we should be able to assess, through formal methods or statistical procedures, the robustness of the trained models and moreover, be able to include in the training methodologies some methods guaranteeing the robustness of the final model.

It is also useful to explore tradeoffs of the training strategies in order to reach an adequate balance between performance and robustness of the resulting ML model.

And last but not least, we should be able to define safety mitigations in the system, if these training and assessing methods are not enough to guarantee the expected safety level.

2.4. Explainability Challenges

The last pillar of AI trustworthiness is the explainability, and some aspects of this challenge for aeronautics have been also described in [CANSO STWG AI, 2021], [NIST,2021], [Arrieta *et al*., 2019], and [Gilpin et *al*., 2018]. Of course, as humans, we need to formulate the reasons why something works or does not work. But this comes back to the black box issue of the ML models, where looking into the model description at software level does not help understanding how it behaves, contrarily to rule-based software. This inherent lack of explainability has triggered active research to increase understanding and confidence in ML-based systems.

The problem is particularly difficult, because the data-based methods only look for correlations while explainability needs causality. We need therefore to develop methods pointing out the causality links between input data and output of the ML modules to express the underlying explanation.

Currently, some of the most performant explanation systems are based also on ML models, which are trained in parallel of the models they need to explain. So, how to gain confidence in the second model and ensure it does not have itself other issues of stability, bias etc.? They can fail in their task, by being trapped into the non-causality or robustness issues. This would lead of course to a completely false explanation and by consequence to lack of trust into the full system. However, very often the failure of the explanation model put in place may highlight some biases in the training data set, so all the challenges (specification, data representativeness, verification and explainability) are actually related to each other.

Another dimension in the explainability is its relation to the human factors. Indeed, one have to consider that the explanation content may differ according to the user who receives the information. . The data scientist, the end user (be it a pilot or an Air Traffic Controller), the regulation authority, they all have different knowledge and different mental models, so they expect explanations for different reasons: the end user needs an almost real time explanation which has a concise form and is pertinent for a decision aid, the data scientist will need a thorough insight into the internal features in order to improve the model, while the accident investigator or regulation authority will need a full file of evidence of the system, but not at all in real time.

Human factors should also be considered for the operational monitoring of AI-based systems, as the Human Machine Interface will give essential indications of anomalies in the functioning of the system. In addition, a feedback should exist to the data management process, since data that led to these behaviors have to be traced back to the root cause and for this they need to be stored during the runtime.

2.4. Risk mitigation Challenge

Finally, an important feature for AI trustworthiness would be the ability to define the limits of safe operation for the ML module, and this is actually the most important characteristic for a safety critical application: the system has to operate only under the situations where it has enough confidence in its own outputs. The system has to operate in its Operational Design Domain only, to guarantee its safe performance.
This relates to system considerations (e.g. monitoring needs, architecture mitigation with safety net), which will be addressed by the future WG-114/G-34 standard, together with the Machine Learning Development Lifecycle explained in the next section.

3. Machine Learning Development Lifecycle

Several attempts of definition of a development lifecycle for machine learning exist in the literature like [ANSI/UL, 2020], [Bhattacharyya et al., 2015], [Cluzeau et al., 2015], [DEEL, 2021], [Hawkins et al., 2021], [Redman et al., 2020], [Wilkinson et al., 2016]. However, these proposals of lifecycle are not fully integrated in the aeronautical standardization framework recalled in the introduction, and most of them have not the sufficient level of details to support certification/approval process in the aeronautical field. The Machine Learning Development Life Cycle (MLDL) defined by WG-114/G-34 is applicable to offline ML technologies. The MLDL does not impose a specific development process (e.g., Waterfall, V, W, or Agile cycle) nor a specific learning environment, nor specific ML technologies. This MLDL is fully integrated with the aeronautical standardization framework and supports the certification/approval process. It also contributes to define and organize development objectives and outputs in a simple and clear manner for certification/approval applicants. Figure depicts the proposed Machine Learning Development Lifecycle.
System or subsystem aspects are not discussed in this section, as they can mostly be addressed with existing standards, even if some adaptation are currently being discussed within WG-114/G-34.

3.1 MLDL inputs

The MLDL has the following inputs: (i) System/subsystem requirements allocated to ML items, and (ii) System/subsystem architecture.
These requirements and architecture are the result of system/subsystem design. These requirements describe the specific function that the Machine Learning items should implement as well as the safety, performance, and other requirements that the Machine Learning items should achieve.

3.2 MLDL outputs

The MLDL processes are complete when their objectives and the objectives of the integral processes associated with them are satisfied. The primary outputs of the MLDL are: (i) a set of ML implementation requirements, (ii) a Test Dataset that should be used in the Software/Hardware item development phase, and (iii) documentation. Documentation will include descriptions of model and data processing as well as results using the training, validation, and test data sets.

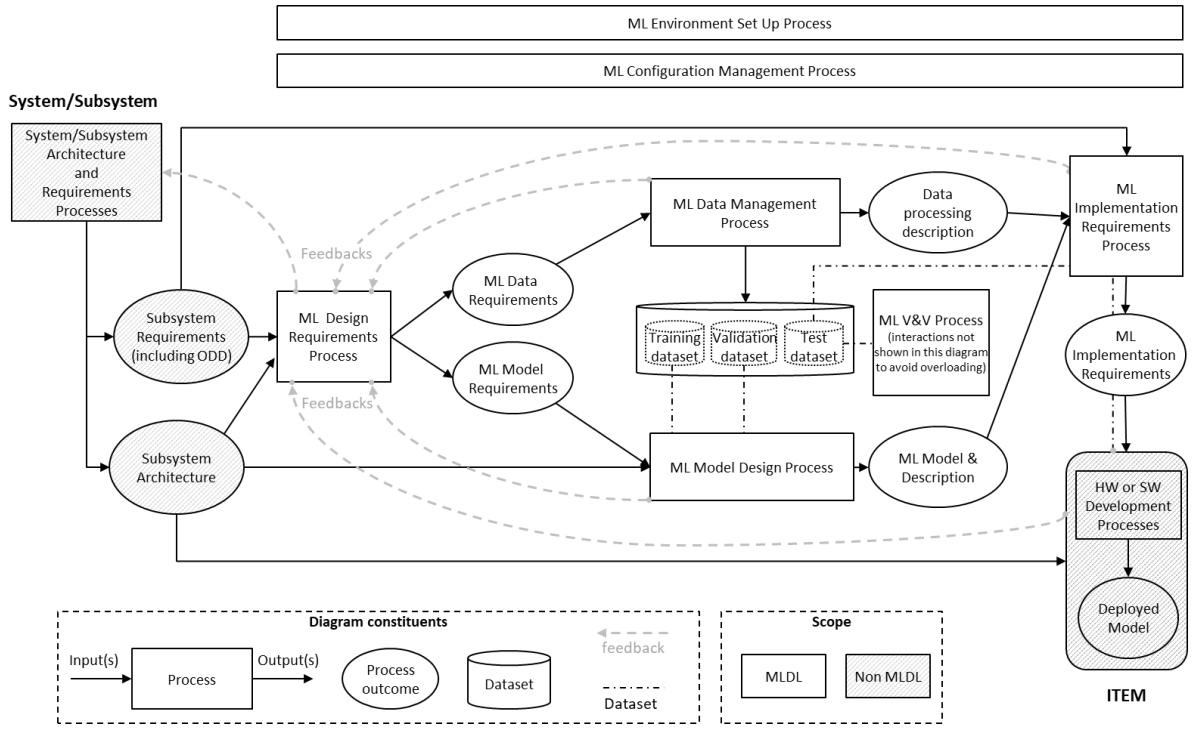

Figure 2 MLDL overview

3. 3 MLDL processes

The MLDL is made of several processes to produce the MLDL outputs.

ML Design Requirements process

This first process aims at refining the system/subsystem requirements allocated to the ML items, in order to develop the ML design requirements.

This process also consolidates the ML derived requirements, i.e. the additional requirements emerging from design or implementation decisions during the ML development process, which are not directly traceable to system/subsystem requirements.

This process takes as inputs the system/subsystem requirements, the system/subsystem architecture, and the feedbacks from the subsequent activities.

This process produces two types of ML design requirements:

ML data requirements (e.g., data sources, data quality objectives, datasets size)

ML Model requirements (e.g., performance metrics, size, type of model, expected properties)

ML Environment Set Up Process

The ML learning environment is an integrated set of methods, tools, and hardware resources used to specify, build, train, optimize, verify, describe, monitor, and reproduce the ML Model. The ML learning environment should be identified from the analysis of ML Model requirements, the ML data requirements, the System/subsystem Architecture and the ML datasets.

ML Data Management process

The ML data management processes aim at producing the datasets used for the ML Model training and verification. The output of these processes is three datasets and the ML input data preparation description:

The training dataset, used to train the ML Model.

The validation dataset, used to tune the hyper parameters.

The test dataset, used for verification activities. Note that the test dataset is not normally used to build (train) or refine (tune) the model.

ML data processing description: this describes the pre-processing of raw data to create the inputs to the model when it is implemented. This may include cleaning, transformation, feature extraction, filtering, range checks, or other data quality checks. Note that this is a description of the processing used to create model inputs in real-time and while it will be informed by, and consistent with, the process used for creating the offline model training data, it could be different because it will be for processing streaming data and not batch data.

Data management relies on several processes, such as data source identification, data collection, data preparation, data allocation, data validation and verification, data maintenance, and data configuration management.

ML Model Design process

The ML Model design process develops the ML Model architecture and identifies the learning environment. It also develops, through one or more iterations, ML training requirements that can be used to build, train, and optimize the ML Model. The outcome of the ML Model design process is the ML Model Description which includes the ML Model architecture and the ML Model implementation requirements. This process is made of several activities:

ML Model architecture development: The ML Model architecture development activity process defines the ML Model architecture including the breakdown into ML Model elements (data pre/post processing, a model or several submodels, model pre/post processing), the description of these individual elements, and how they should be integrated to comply with ML Model requirements. The ML Model architecture does not define the design of the

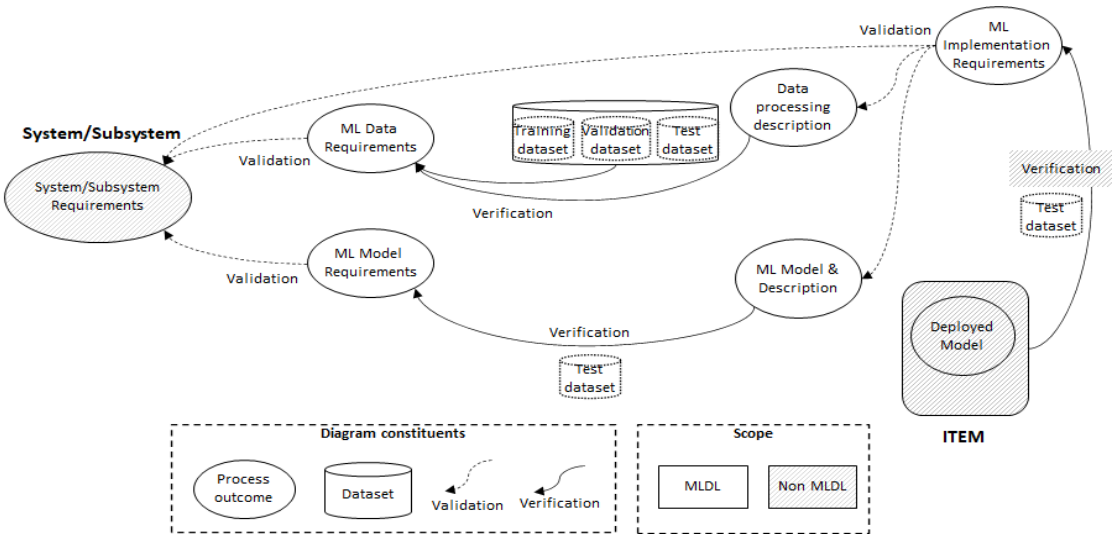

Figure 3 MLDL Validation and Verification activities

ML model elements since it is done by the ML Model building activity.

ML Model training specification: The ML model training specification activity develops ML training requirements that can be used to build, train, and optimize the ML model. Training requirements should contribute to ML Model explainability and reproducibility.

ML Model building: The ML Model building activity develops appropriate set of hyper parameters for a candidate ML model. The ML Model building activity consists of the following steps: (i) the ML Model hyper parameters are selected and if needed optimized from the ML Model architecture to comply with the ML Model requirements and if applicable with the ML training requirements, and (ii) the ML Model is developed from the hyperparameters defined in the ML Model architecture and if applicable in the ML training requirements.

ML Model training: The analytical form of the model, and the hyperparameters are known to the ML designer at the end of the ML Model Building activity while the weights are unknown to the designer unless a previously designed similar model is being used as a starting point. The training phase aims at developing/computing the weights using the ML model optimization algorithm to reach the expected performance of the model on training dataset.

ML Model Post-Training Optimization Activity: Model Optimization Activity consists in performing changes after the training phase to the candidate ML Model to achieve the expected performance specified in the ML Model Requirements.

ML Model description: The ML Model description activity develops the sufficient documentation to facilitate the specification of ML Implementation requirements.

ML Implementation Requirements process
The ML Implementation Requirement process develops a baseline of ML Implementation requirements that can be used to implement the ML Model and its associated data processing on the target computer(s). Sub-system implementation constraints are also included in this baseline of ML Implementation requirements.

ML Validation and Verification processes
In order to support development assurance, the MLDL is subject to validation and verification activities. In the context of the MLDL, validation and verification have the following meanings:

Validation: the determination that the ML requirements are correct and complete

Verification: the evaluation of the ML model to determine that it meets the ML requirements

Fig. 3 summarizes the validation and verification activities performed on the different outcomes of the MLDL processes. Because validation and verification activities are performed on the outcomes and not on the processes, Figure focuses on the outcomes (depicted with bubbles) and not on the processes (depicted with rectangles).
Validation is performed on ML Data Requirements, ML Model Requirements, and ML Implementation Requirements. Verification is performed on Training, Validation and Test datasets, Data processing descriptions, ML Model (using the Test dataset). Validation of System/subsystem Requirements and verification of the deployed model are out of the scope of the MLDL.

ML Configuration Management Process
The ML Configuration Management process, working in cooperation with the other life cycle processes, provides a defined and controlled configuration of the ML datasets and ML Model throughout the MLDL. It provides among others: (i) the ability to consistently replicate the ML datasets and the ML Model for the implementation phase or to regenerate them in case of a need for investigation or modification, (ii) control of process inputs and outputs that ensures consistency and repeatability of process activities, (iii) a known point for review, assessing status, and change control by the establishment of baselines, (iv) controls that ensure problems receive attention and changes are recorded, approved, and implemented, (v) evidence of approval of the ML Model by control of the outputs of the MLDL processes, (vi) the assessment of the ML Datasets and ML Model compliance with requirements, and (vii) maintenance of secure physical archiving, recovery, and control for the configuration items.

4 Outlook

WG-114/G-34 has just published a Statement of Concerns on AI for Aeronautical Systems (EUROCAE ER-022 / SAE AIR6988). This study addresses several aspects. It starts with a taxonomy of the AI techniques, followed by a gap analysis with the current standards. This allowed to identify and characterize the main areas of concerns, and then some possible leads to address them. Last sections deal with airborne and ground operations use cases that were collected in the effort to represent the industry's needs. The Statement of Concerns outcomes shaped the main assumptions of the WG-114/G-34 objectives. First, the scope of the standard was reduced to the ML technique and in particular to the offline ML (learning does not continue during operations). Second the joint group was structured into 7 subgroups (SG) to map the ML development workflow but not only: SG1: Use cases management; SG2: Data management and validation; SG3: ML design and verification; SG4: ML implementation and verification; SG5: System & safety considerations; SG6: Change of previously developed ML systems; SG7: All other integral processes.
The first issue of the standard is scheduled to release by mid-2023. This is a very ambitious timeframe for a standard that is cross-domains (airborne and ground) and for a technology that is relatively new to the industry or even still part of the research field. The first issue will be reduced to the offline ML technique, then in a second issue other AI technologies will be addressed.

References

[ANSI/UL, 2020] ANSI/UL, *UL 4600 Safety Standard for Autonomous Vehicles*, ANSI/UL standard, 2020.

[Bhattacharyya et *al*., 2015] Siddhartha Bhattacharyya, Darren Cofer, David J. Musliner, Joseph Mueller, and Eric Engstrom, *AFE-87 Certification Considerations for Adaptive Systems*, NASA/CR–2015-218702, 2015.

[Cluzeau et *al*., 2015] Jean Marc Cluzeau, Xavier Henriquel, Georges Rebender, Guillaume Soudain, Luuk van Dijk, Alexey Gronskiy, David Haber, Corentin Perret-Gentil, Ruben Polak, *Concepts of Design Assurance for Neural Networks (CoDANN)*, EASA Public Report Extract, 2020.

[DEEL, 2021] DEEL Certification Workgroup, *Machine Learning in Certified Systems*, DEEL White Paper, 2021.

[Hawkins et *al*., 2021] Richard Hawkins, Colin Paterson, Chiara Picardi, Yan Jia, Radu Calinescu and Ibrahim Habli, *Guidance on the Assurance of Machine Learning in Autonomous Systems (AMLAS)*, Assuring Autonomy International Programme, 2021.

[Redman et *al*., 2020] David Redman, Donald Ward, Matt Carrico, AFE 87 project members, *AFE-87 Machine Learning Final Report*, AVSI document, 2020.

[Wilkinson et *al*., 2016] Chris Wilkinson, Jonathan Lynch, Raj Bharadwaj, and Kurt Woodham, *Verification of Adaptive Systems*, DOT/FAA/TC-16/4 document, 2016.

[ED-12C/DO-178C, 2011] EUROCAE/RTCA, *Software Considerations in Airborne Systems and Equipment Certification*, 2011

[ED-218/DO-331, 2011] EUROCAE/RTCA, *Guidance for Model-Based Development and Verification*, 2011

[ED-109A/DO-278, 2011] EUROCAE/RTCA, *Software Integrity Assurance Considerations for Communication, Navigation, Surveillance and Air Traffic Management (CNS/ATM) Systems*, 2011

[ED-153, 2009] EUROCAE, *Guidelines for ANS Software Safety Assurance*, 2009

[ED-80/DO-254, 2000] EUROCAE/RTCA, *Design Assurance Guidance for Airborne Electronic Hardware*, 2000

[ED-79A/ARP4754A, 2011] EUROCAE/SAE, *Guidelines for Development of Civil Aircraft and Systems*, 2011

[2017/373] *Commission Implementing Regulation (EU)*, 2017

[ER-022/AIR6988, 2021] SAE G-34/EUROCAE WG-114, *Artificial Intelligence in Aeronautical Systems: Statement of Concerns*, 2021

[CANSO STWG AI, 2021] Béatrice Pesquet et al., *CANSO Strategic Technology Workgroup (STWG) White Paper on "Artificial Intelligence – Enablers and Use Cases"*, 2021

[NIST, 2021] https://www.nist.gov/artificial-intelligence/ai-foundational-research-explainability

[Gupta et al., 2021] Kavya Gupta, Jean-Christophe Pesquet, Beatrice Pesquet-Popescu, Fateh Kaakai, *A Quantitative Analysis of the Robustness of Neural Netwroks for Tabular Data*, IEEE ICASSP, June 2021

[Katz et al., 2017] Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer, "*Reluplex: An efficient SMT solver for verifying deep neural networks*," in International Conference on Computer Aided Verification. Springer, 2017, pp. 97–117.

[Combettes et al., 2020] Patrick L Combettes and Jean-Christophe Pesquet, "*Lipschitz certificates for neural network structures driven by averaged activation operators*," SIAM Journal on Mathematics of Data Science, vol. 2, pp. 529–557, 2020.

[Latorre et al., 2020] Fabian Latorre, Paul Rolland, and Volkan Cevher, "*Lipschitz constant estimation of neural networks via sparse polynomial optimization*," arXiv preprint arXiv: 2004.08688, 2020.

[Gilpin et *al*., 2018] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, '*Explaining Explanations: An Overview of Interpretability of Machine Learning*', ArXiv180600069 Cs Stat, May 2018.

[Arrieta *et al*., 2019] A. B. Arrieta *et al*., '*Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges toward Responsible AI*', *ArXiv191010045 Cs*, Dec. 2019, Accessed: Mar. 09, 2020.

[Rhie et *al*., 2017] Y. L. Rhie, J. H. Lim, and M. H. Yun, '*Evaluating Representativeness of Qualitative Text Data in Identifying UX Issues*', Int. J. Human-Computer Interaction, vol. 33, no. 11, pp. 868–881, 2017.

[Gupta et al., 2021b] Kavya Gupta, Jean-Christophe Pesquet, Beatrice Pesquet-Popescu, Fateh Kaakai and Fragkiskos Malliaros, *An Adversarial Attacker for Neural Networks in Regression Problems*, IJCAI Workshop on Artificial Intelligence Safety (AI Safety), 2021