AI Assurance for the Public -- Trust but Verify, Continuously

Phil Laplante
Engineering
Penn State
Malvern, USA
plaplante@psu.edu

Rick Kuhn
Computer Security Division
NIST
Gaithersburg, USA
kuhn@nist.gov

***Abstract*— Artificial intelligence (AI) systems are increasingly seen in many public facing applications such as self- driving land vehicles, autonomous aircraft, medical systems and financial systems. AI systems should equal or surpass human performance, but given the consequences of failure or erroneous or unfair decisions in these systems, how do we assure the public that these systems work as intended and will not cause harm? For example, that an autonomous vehicle does not crash or that intelligent credit scoring system is not biased, even after passing substantial acceptance testing prior to release.**

In this paper we discuss AI trust and assurance and related concepts, that is, assured autonomy, particularly for critical systems. Then we discuss how to establish trust through AI assurance activities throughout the system development lifecycle. Finally, we introduce a "trust but verify continuously" approach to AI assurance, which describes assured autonomy activities in a model based systems development context and includes post-delivery activities for continuous assurance.

Keywords—artificial intelligence, zero trust, explainable AI

I. Introduction

Artificial intelligence (AI) systems are increasingly seen in many domains that interact directly with the public, such as self -driving delivery and passenger vehicles, autonomous maintenance equipment (e.g. vacuums and lawnmowers), and medical systems (e.g. expert diagnosis or robotic surgery). AI systems should equal or surpass human performance, but given the consequences of failure in these systems, how do we gain and maintain the support of the public that these systems are safe?

In a 2021 poll of 2200 American adults conducted by Stevens Institute of Technology, 48% of respondents felt that the positives of greater AI adoption in everyday life outweighed the negatives [1]. But 29% believe the opposite that the negatives outweighed the positive. A majority of respondents also believe that in the future AI should play a greater role in technology, manufacturing, logistics and retail applications. It seems that the public wants to trust AI systems, but this trust is subject to continuous reevaluation and could be withdrawn if a significant disaster or series of disasters further expose the risks. Therefore a trustworthy, widely applicable and repeatable approach for AI assurance will be an important driver for adoption and continued support of AI systems by the public.

II. Trust in AI

Trust in an autonomous system has been defined as "the degree to which there is confidence that the system will behave as intended"[2]. Some would say "the system could pass the Turing test" – that is, the system should behave like a rational human. We suspect it will be demanded that autonomous systems work much better than humans doing the same task. For example, an accident rate associated with self-driving cars that is as high as human traffic accident rates would probably bankrupt the carmaker. Unfortunately, the tort and liability issues are as challenging as the technical ones.

There are several contributors to trust that are a superset of the key elements of AI assurance. These include:.

- Reliability¸
- Safety¸
- Security,
- Privacy,
- Availability,
- Usability, and
- Explainability.

Consistent with the goal of AI assurance, trust building is a dynamic process that needs to be established at the outset of system use and maintained continuously. We therefore seek an AI system development and assurance process that both establishes trust at the outset and maintains that trust over long periods of usage of the system.

III. What is AI Assurance?

AI assurance is a process that is applied at all stages of the AI engineering lifecycle to ensure that any intelligent system is producing outcomes that are valid, verified, data–driven, trustworthy and explainable to the layman, ethical in the context

978-1-6654-8864-8/22/$31.00 ©2022 IEEE
DOI 10.1109/STC55697.2022.00032

of its deployment, unbiased in its learning and fair to its users [3].

An important question related to assurance is: how do we assure that AI systems are safe in a complex and rapidly changing environment when conventional test coverage and formal verification methods are insufficient or cannot be applied? Achieving assurance of function in any environment requires showing that the test environments adequately cover real-world conditions that may be encountered. Although some statistical and structural coverage metrics are relevant they are usually Inadequate for many of the challenges in autonomous systems assurance. AI assurance also involves answering the question: how do we determine that the data gathered to train an AI system is suitably representative of the real world?

Another natural question to ask is: what kinds of guidance can we get from standards for AI assurance trust and so on with respect to AI testing and assurance? A partial answer is there are many standards related to AI including several IEEE sponsored AI assurance standards that are either developed or in development and include standards for transparency, bias, and ethics; including:

- IEEE P7001 - Transparency Of Autonomous Systems,
- IEEE P7003 - Algorithmic Bias Considerations,
- IEEE P7006 – Personal Data AI Agent Working Group,
- IEEE P7007 – Ontological Standard for Ethically driven Robotics and Automation Systems,
- IEEE P7008 - Standard for Ethically Driven Nudging for Robotic, Intelligent and Autonomous Systems.
- IEEE P7009 - Standard for Fail-Safe Design of Autonomous and Semi-Autonomous Systems,
- IEEE Std 7010 – Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being,
- IEEE P7014 – Standard for Ethical considerations in Emulated Empathy in Autonomous and Intelligent Systems.

Most of these standards mention the importance of testing and assurance but give little operational guidance. Clearly, more standards are needed.

IV. Critical versus Noncritical AI

There is a difference between AI used in critical infrastructure systems versus non-critical systems and this difference probably impacts public trust. Table I shows application domains for various kinds of critical systems and some examples of those systems.

TABLE I. Application domains for critical systems, system examples and possible uses for AI (adapted from [4]

Application Domain	System Examples	Typical AI Uses
Telecommunication infrastructure	Public telephone network, local branch exchange	ID, FA
Water supply systems	Water treatment plant, dam control	FA, FS, PHM
Electric power systems	Nuclear power plant, regional electrical grid	FA, FS, PHM, PO
Oil and gas generation and distribution	Gas pipeline, gas-powered power plant	FA, FS, PHM, PO
Roadway transportation systems	Smart interstate highway, traffic monitoring and control	FA, FS, PHM, PO
Railway transportation systems	High-speed train line, metropolitan train network control	FA, FS, PHM, PO
Air transportation systems	Air traffic control system network, passenger aircraft autopilot	FA, FS, PHM, PO
Banking and financial services	Pension fund management, stock market management	FD, ID
Public safety services	Air passenger screening, police dispatch	FD, ID, PO
Health-care systems	Robotic surgery, health record management	FD, ID, PO
Administration and public services	Employee personnel database, retirement management	FD, ID
FA: failure analysis; FD: fraud detection; FS: fail-safe operation; ID: intrusion prevention and detection; PHM: system prognostics and health management; PO: performance optimization.		

The far right column of Table I indicates some of the ways that AI can be used in those systems, for example, in failure analysis, fault detection, and so on [4].

AI is also used in many noncritical applications for example in entertainment, shopping, and convenience based systems. But even though noncritical applications may seem harmless they could interact with critical systems or could be through misuse or abuse become dangerous to the user. Therefore, it is important to consider AI assurance for every system that utilizes AI.

To further the discussion of AI assurance issues going forward, consider three classes of consumer-facing AI systems. The first class consists of robotic convenience systems, for example, AI based delivery, vacuum cleaners or lawn cutting systems. The second class of systems comprise autonomous vehicles, such as self-driving cars and delivery vehicles. The third class of AI systems includes various AI guided health care systems, for example for disease detection, diagnosis and treatment planning. We will refer to these classes or instances going forward.

V. Critical versus Noncritical A Hazards and Failure in AI Systems

Bad things can happen to AI enabled systems, which is probably why the public is so wary. First, there are various types of adversarial attacks that can degrade or corrupt system performance. Examples include:

- Trojan or backdoor attacks involving injection of certain data to the AI system causing it to give a specific incorrect response.
- Model inversion, i.e. extracting private or proprietary model information from the AI system by strategic introduction of data.
- Tampering, i.e. causing AI system to fail by providing false data, e.g. by obscuring cameras or injecting corrupted data into collection points.

The AI system could also experience a "payload" degradation, which can result in decreased performance or increased errors in the outputs of the system. Or there could be a mission change say from a system that was designed for military purposes being converted to civilian use, or vice versa, which could lead to unwanted even dangerous consequences.

Consider the exemplar systems previously introduced -- all kinds of bad things could happen to them. For example, a robot vacuum cleaner could get trapped behind a door or tangled in the carpet. It could get snagged on a lamp cord, toppling the lamp and starting a fire. It could run over something dirty, sticky or oily making a mess or slip hazard. It could also suck up something valuable or important like a diamond earring or contact lens. The autonomous vehicle could drive on the wrong side of the road or fail to avoid a collision. The medical diagnosis system could miss important symptoms that could produce an incorrect diagnosis or lead to a faulty treatment plan.

Here is where AI assurance is paramount because the public needs to know that these kinds of threats have been anticipated and precluded via deliberate design, by testing and by continuous external monitoring or self-monitoring.

VI. Key Aspects of AI Assurance and Goals

According to the NIST AI Risk Management Framework, trustworthy AI is "valid and reliable, safe, fair and bias is managed, secure and resilient, accountable and transparent, explainable and interpretable, and privacy-enhanced [5]. Explainability is a key aspect of AI validation [6].

The need for explainability and transparency can be driven the distrust in AI systems created by potential system bias. In social domains, for example, healthcare or finance, bias may result in unfair and unethical decisions.

In a NIST sponsored workshop participants identified nearly 40 types of AI bias including behavioral, exclusion, historical and institutional [7]. Some examples of bias in technical domains include an autonomous vehicle driving on the wrong side of the road, a robot floor cleaner setting the height incorrectly and a disease diagnosis algorithm detecting one particular disease too often. Whether we are concerned with social or technical domains, a common source for the problem is that the AI was not trained on an appropriate set of data.

In addition to the use of training datasets and/or practices that are inadequate or inherently biased or non- representative, bias problems may result from lack of testing, and deployment of technology that is either not fully tested, potentially oversold, or based on questionable or non-existent science. Bias can also include statistical anomalies such as overfitting or underfitting or due to skewing or incomplete data in the environment.

Bias can be identified and mitigated through a number of means including data collection best practices, analysis of contextual awareness, statistical measures, analysis of variance, outlier detection, causal inference, and many other techniques. The workshop participants noted that bias is not unique to AI systems, and that while it is impossible to eliminate bias completely, the goal in AI assurance is to find ways to identify, understand, measure, manage and reduce bias. To achieve these goals industry specific standards and guidelines can help. An overarching issue in bias problems is the need for a rigorous formal definition of what is meant by 'bias', in order to measure and detect it. Different users or organizations may have different definitions, so a necessary step is an agreed upon definition for the particular field of use, or specific application, where the AI will be used.

VII. Model Based Systems Engineering

Now we examine a process model for AI assurance. There are many that could be used but we chose Model-Based Systems Engineering. According to INCOSE Model-based systems engineering (MBSE) is "the formalized application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases.". MBSE focuses on domain models rather than on document- based information exchange. MBSE's effectiveness is largely based on the effectiveness of coupling models, machines and teams for increased collaboration, improved communication and shared understanding.

MBSE makes systems engineering more rigorous, precise and repeatable through the use of multiple, interconnected systems models. These models support system requirements, design, analysis, verification and validation, from the conceptual design phase and continuing throughout development and later life- cycle phases. MBSE is well-known to manage complexity, provide quality and productivity improvements and lower risk through use of rigor and precision fostering better communications between the development team and customers. Model Based Systems Engineering is widely used all over the world and it's an excellent system development paradigm for AI system assurance, which we will further elaborate.

MBSE includes models for:

- requirements, hardware and software architectures,
- design elements (including hardware drawings and software design documents),
- embodiments of those designs (including circuit diagrams and code),
- test cases from unit tests through acceptance tests (manual and automated),
- and maintenance documentation and information.

In all cases these models provide opportunities for explainability and transparency.

To further the understandability of these models, each should use a domain appropriate mix of natural languages, graphical notations, formal syntax and semantics. Storing these under strict configuration management helps to preserve the integrity and trust in these models. Some MBSE notations and design languages are sufficiently rigorous for formal specification, and could be effective in achieving AI assurance.

VIII. MSBE and V&V

The classic V model is often used in MBSE because it recognizes that verification and validation is a done throughout the systems lifecycle (Fig. 1).

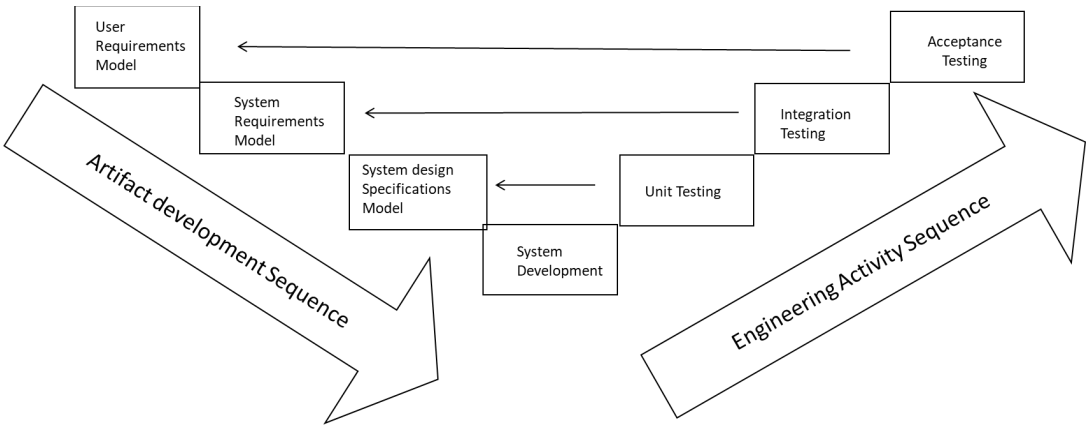

Fig. 1. Classic V model for systems engineering

The V model closely ties the forward engineering sequence of modeling and development activities with the downstream activities that provide for verification and validation.

In Fig. 1 we start with requirements modeling on the far left– which we already noted is very important for capturing context along with all other required behaviors of the AI system. The model created in this process is tested in the acceptance testing phase, shown on the right, just prior to delivery. This is both a verification and validation activity as validation for the user requirements has to occur early. We will discuss some techniques for this shortly.

A similar process occurs for the system requirements and design specification models – which embody the architectural and detailed design of the system. These models are verified through integration testing and unit testing respectively. These occur in reverse order after components of the system have started to be built. But validation of the system requirements and system design models have to occur at the time of their development. The V-model represents one paradigm for systems development, but in our case we'll use it as a framework to discuss appropriate lifecycle AI Assurance activities.

A. Requirements V&V Techniques for AI Assurance

There are many techniques that are appropriate for requirements elicitation, verification, and validation of complex systems. But for AI systems, a smaller subset of techniques is also appropriate. These techniques include might include systematic manual analysis of the requirements, group reviews and inspections, for example, those found in the Joint Application Development method and Quality Function Deployment (QFD). Prototyping – both executable and nonexecutable is also appropriate. Using an executable model of the system test-case generation (for testability and completeness) are also recommended. Both automated consistency analysis and other formal methods (for example, for model and consistency checking) are highly appropriate. Finally, viewpoint resolution and task analysis (often via user stories and use cases) are highly recommended. Of course the mix of techniques to be used depends on many factors and must be carefully chosen.

B. Design V&V Techniques for AI Assurance

Many traditional techniques are recommended for both high-level architectural and detailed design of AI systems. These include large-scale requirements engineering approaches such as Joint Application Development. Continuous prototyping, including the kind used in agile and lean agile development methods are also useful. In Quality Function Deployment (QFD) the "house of quality" can be used to compare alternative designs. QFD emphasizes the "what's" and "how's" which fosters a high level of explainability and transparency.

Designer as apprentice is another widely used technique that can be used to uncover the details when automating human behaviors. Finally, rigorous consensus building techniques such as wideband Delphi and the analytical hierarchy process (or AHP) can be used to provide transparency for and to validate and verify different design choices.

C. Testing for AI Assurance

When it comes to testing in the AI assurance world, there needs to be a strong focus on software testing. Even though many of the underlying AI and machine learning algorithms can be implemented in firmware or hard-wired, the same testing techniques can be used.

Referring back to the V model, we can divide the testing types into unit, integration and acceptance testing. Traditional software testing includes various techniques for unit testing, subsystem testing, system testing and acceptance testing. But traditional approaches do not necessarily address the unique needs of AI testing. Menzies suggests that AI systems testing should include static analysis, runtime verification and model checking [8]. Another problem associated with testing AI systems is: do you have enough data and the right data for training? Testing should be focused on answering this question.

But greater questions lurk for AI assurance. Namely, how much and what kind of unit testing is needed to reach acceptable levels of assurance? How much testing coverage is needed, for

example, All-Paths? How much integration testing is enough? How much acceptance testing is enough? The unique nature of opaque neural networks and the inapplicability of traditional test adequacy criteria loom here and answers to these questions can only be found by careful examination of the system objectives, context, application domain and so on.

D. Dealing with Uncertainty

A well-known problem in AI systems is that they do not always deal well with uncertainty. For example what does the robot vacuum cleaner do when encountering an object of unknown type, for example, something sticky? What does the autonomous vehicle do if it somehow veers off its navigational map? And how does the medical diagnosis system deal with symptoms that are not in its database?

There are many frameworks for dealing with uncertainty including: probabilistic reasoning, possibility theory, Dempster-Shafer theory, fuzzy logic, rough sets, intelligent agents, captured expert opinion and many more [9].

The problem of uncertainty needs to be dealt with in every AI system or else high levels of assurance are not achievable.

IX. Special V&V Considerations for AI Assurance

We have already noted that conventional V&V is not enough. Testing for AI should include appropriate kinds of traditional testing plus testing to address explainability, transparency and bias. So it stands to reason that conventional approaches to V&V at each stage are not enough. We need additional tools. Two tools that can help address these special problems are combinatorial testing and formal methods.

A. Combinatorial Coverage Measurement for AI

The main difference between conventional SW and AI algorithms is that the latter are often opaque box functions, where only the input changes. For example, with neural nets of all types, performance is dependent on the input training set, which is used in establishing the connections and weights in the network. Conventional structural coverage measures, such as branch or condition coverage, have little practical value in these problems, because the code for the neural net processing is the same across applications, only the connections and weights are different. Therefore, measures of input space coverage are needed, to show that the input set used to train the AI/ML system is sufficiently representative of objects and environments that will be seen in practice.

Measurement methods derived from combinatorial testing can be used to evaluate the degree to which combinations of input parameter values are included in a training set [10], [11]. These methods are related directly to attributes of the training set that are significant for AI, because they measure the level of inclusion of every t-way combination of input values. Combinations of input values are used either directly or indirectly in most AI systems, for example the inclusion or absence of various trait combinations are the basis for classifying animals and plants into different species, and consequently are essential in AI processing of images or property databases.

Combinatorial methods are also very effective for explanations and justifications of decisions in AI/ML systems, which are essential not only in operation but also in validation phases of assurance. If the AI behavior cannot be explained, then it will be difficult to trust its conclusions, and if the training inputs do not adequately represent the real environment, then we cannot have confidence that it works correctly across all inputs that may occur in practical use.

B. Formal Methods

Formal methods are necessary for V&V of AI systems but not sufficient, and others agree with this assessment [2]. Formal methods can be used throughout the system lifecycle, for example, to write parts of the requirements specifications using a formal notation, or to apply formal proving techniques to various representations of the design and implementation of the system, which fosters a high degree of explainability, transparency and can expose bias.

Formal proving techniques can be used to validate those specification. Other kinds of automated consistency analysis or theorem proving can be used at various times. The formality of the models helps communicate with domain experts more concisely. In some cases semantics-preserving transformations can be used to convert the specification directly into code. And formal methods can be used for some verifications, for example to verify that the implementation matches the specification (testing).

SysML comprises a set of modeling languages that are an essential enabler to MBSE. SysML is essentially a subset of the UML with some extensions. SysML can be classified as semiformal since all of its meta models have precise mathematical equivalents. These need to be augmented with semantics in order to enable their use for formal analysis of specifications.

X. A "Trust but Verify" Process Model

Wing notes that "The set of trustworthiness properties for AI systems, needs to be extended beyond reliability, security, privacy, and usability to include properties such as probabilistic accuracy under uncertainty, fairness, robustness, accountability, and explainability" [2]. This observation, which has been consistent with many of the ideas presented in this white paper, are reminiscent of the zero trust security architecture.

The Zero-trust cybersecurity architecture was introduced by Forrester Research in 2010, and it means that the system is continuously evaluated for security properties. That is there is no such thing as testing the system, deploying it and assuming it is always secure.

The same has to be done with respect to AI assurance. Even after robust verification and validation for all of the key assurance properties, the system must never be regarded as always safe. The system must be continuously evaluated and challenged. This is reminiscent of the famous "trust but verify" observation attributed to many individuals, which has a more encouraging ring to it for the public than "zero trust" (though in reality, this is a zero trust AI). Therefore we choose to use the former phrase, not the latter.

XI. "Trust but Verify" AI Systems Development Model

Now consider the "Trust but Verify" AI systems development model shown in Fig. 2, which is adapted from a suggestion found in [12].

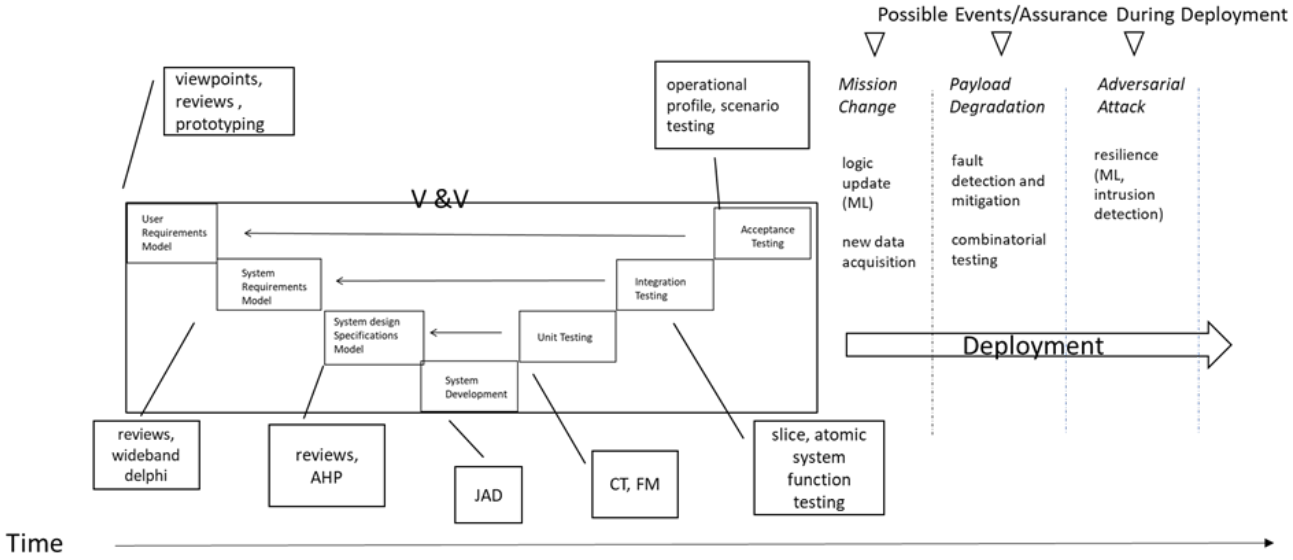

Fig. 2. AI Assurance lifecycle development model

It consists of the traditional V model where the call out boxes indicate a few of the assurance activities at each phase of the lifecycle, which we have already discussed as being appropriate for AI systems. For example, using viewpoints and various kinds of prototyping for user requirements elicitation and various kinds of reviews at other phases of the lifecycle.

To the traditional V systems development model we add continuous assurance activities which occur after system deployment. These activities are conducted in response to certain events over the lifespan of the AI system. For example, if the system mission or context changes, then machine learning algorithms will need to be reevaluated and possibly updated.

Through continuous monitoring of the system, it may be determined that performance is degrading. In this case, new fault detection and mitigation algorithms may need to be deployed, and combinatorial testing may be needed to determine the weak spots in the existing algorithms. Finally, through continuous monitoring if it is determined that some form of attack has occurred then new, more resilient machine learning algorithms and enhanced intrusion detection monitoring may be needed.

XII. Continuous Monitoring – A "Trust but Verify Continuously" AI Architecture

Continuous monitoring of the AI system is an important part of the post-deployment assurance process model just described. It therefore makes sense that this continuous self-monitoring would be built into the system. Embedded critical systems use built-in self-testing, that is, real-time periodic self-diagnostics to monitor critical hardware components and report failures. We expect self-diagnostics to be performed in critical AI systems to be performed at start up and continuously during operations.

But internal self-diagnoses is not trustable because of self-gaslighting – if a system reports that a component or algorithm hst but verify" AI architecture.

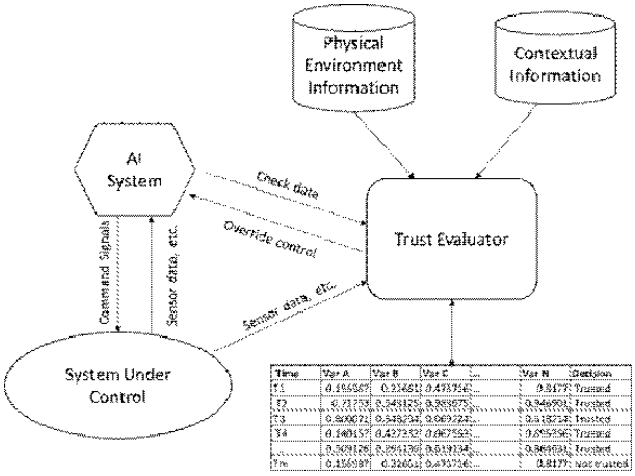

Fig. 3 Model "Trust but Verify Continuously" AI Architecture

In this architecture the trust evaluator has external access to the AI system controlling the critical system. The trust evaluator could be another AI system, a human monitor, a panel of human monitors or some other kind of system. The trust evaluator continuously monitors the inputs to and outputs from the AI system, as well as those to and from the system under control. The trust evaluator also processes contextual information from the real world, including legal and political factors. A decision

table structure is updated with time-stamped values of certain parameters (variables) from system under control, AI control system, environment and context the trust decisions made by the trust evaluator at that time. This information is used for reflection by the trust evaluator and for later analysis including post mortem analysis (black box recording analysis). The Collins/Kestral runtime assurance architecture is an instance of this approach and they have demonstrated its effectiveness on a collision avoidance use case [13].

A primary function of the trust evaluator is to act as a referee that can stop the system at any time if it exceeds some trustworthiness boundaries, for example, making more than n>1 minor or one major decision trust violation. The question of what certifications/evaluations there needs to be for the trust evaluation system (in whatever form) is an open one.

XIII. CONCLUSIONS

In this paper we discussed the issue of trust and AI assurance in public facing systems, especially those that are critical. We have covered some of the activities that need to occur at each stage of the development lifecycle, and we have introduced a "trust but verify continuously" philosophy and architecture for AI assurance.

We have raised more questions pertaining to AI assurance than we have answered. But on aggregate, AI assurance relates to the standard approach of assuring the system development process, the system itself, and the people building the system. We have discussed the process and product but have deliberately left out the discussion of the people involved. That discussion involves setting forth the necessary education, certification/licensing and background checking for all involved and is a discussion for another day. Further development of best practices for AI assurance that incorporate trust but verify continuously are also needed.

REFERENCES

[1] J. Corso, "Americans' Attitudes Toward AI," Stevens *TechPulse* Report, https://www.stevens.edu/news/stevens-techpulse-report, 2021.

[2] J. M.. Wing, "Trustworthy AI." Communications of the ACM vol. 64, no. 10, 2021, pp. 64-71.

[3] L. Freeman, A. Rahman, and F. Batarseh. "Enabling Artificial Intelligence Adoption through Assurance" *Social Sciences, vol. 1*0. , no. 9, 2021, p. 322.

[4] P. Laplante, D. Milojicic, S. Serebryakov and D. Bennett, "Artificial Intelligence and Critical Systems: From Hype to Reality," *Computer*, vol. 53, no. 11, 2020, pp. 45-52.

[5] NIST AI Risk Management Framework: Second Draft, August 18, 2022.

[6] NIST, Combinatorial Testing for Autonomous Systems, https://csrc.nist.gov/Projects/automated-combinatorial-testing-for-software/autonomous-systems-assurance/autonomous-vehicles, accessed 8/20/2022.

[7] NIST "Proposal for Identifying and Managing Bias in Artificial Intelligence," SP 2170, https://www.nist.gov/artificial-intelligence/proposal-identifying-and-managing-bias-artificial-intelligence-sp-1270.

[8] T. Menzies and C. Pecheur. "Verification and validation and artificial intelligence." *Advances in computers* vol. 65, 2005, pp. 153-201.

[9] J. George and M. Wierman, "Uncertainty-based information: elements of generalized information theory," vol. 15. *Physica*, 2013.

[10] R. Kuhn and R. Kacker, "An application of combinatorial methods for explainability in artificial intelligence and machine learning (draft).," NIST, May 2019,

[11] E. Lanus, L. Freeman, R. Kuhn and R. Kacker,. "Combinatorial Testing Metrics for Machine Learning," *IEEE International Conference on Software Testing, Verification and Validation Workshops*, 2021, pp. 81-84.

[12] E. Lanus, I. Hernandez, A. Dachowicz, L. Freeman, M. Grande, A. Lang, J.. Panchal, A. Patrick, and S. Welch. 2021, "Test and evaluation framework for multi-agent systems of autonomous intelligent agents," . arXiv, arXiv:2101.10430. 2021.

[13] D. Cofer, "Machine Learning and the Unknown Unknowns," High Confidence Software and Systems Conference Series, virtual, May 16, 2022.