PESARO 2025

The Fifteenth International Conference on Performance, Safety and Robustness in Complex Systems and Applications

ISBN: 978-1-68558-280-7

May 18 - 22, 2025

Nice, France

PESARO 2025 Editors

Mohammad Rajabalinejad, UTWente, Netherlands

PESARO 2025

Forward

The Fifteenth International Conference on Performance, Safety and Robustness in Complex Systems and Applications (PESARO 2025), held between May 18-22, 2025 in Nice, France, continued a series of events dedicated to fundamentals, techniques and experiments to specify, design, and deploy systems and applications under given constraints on performance, safety and robustness.

There is a relation between organizational, design and operational complexity of organization and systems and the degree of robustness and safety under given performance metrics. More complex systems and applications might not be necessarily more profitable, but are less robust. There are trade-offs involved in designing and deploying distributed systems. Some designing technologies have a positive influence on safety and robustness, even operational performance is not optimized. Under constantly changing system infrastructure and user behaviors and needs, there is a challenge in designing complex systems and applications with a required level of performance, safety and robustness.

We welcomed academic, research and industry contributions. The conference had the following tracks:

- Methodologies, techniques and algorithms
- Applications and services

We take here the opportunity to warmly thank all the members of the PESARO 2025 technical program committee, as well as all the reviewers. The creation of such a high quality conference program would not have been possible without their involvement. We also kindly thank all the authors who dedicated much of their time and effort to contribute to PESARO 2025. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions.

We also thank the members of the PESARO 2025 organizing committee for their help in handling the logistics and for their work that made this professional meeting a success.

We hope that PESARO 2025 was a successful international forum for the exchange of ideas and results between academia and industry and to promote further progress in the areas related to performance, safety and robustness in complex systems. We also hope that Nice provided a pleasant environment during the conference and everyone saved some time to enjoy the historic charm of the city.

PESARO 2025 Chairs

PESARO Steering Committee
Wolfgang Leister, Norsk Regnesentral (Norwegian Computing Center), Norway
Mohammad Rajabali Nejad, University of Twente, the Netherlands
Rémy Houssin, Université de Strasbourg - ICube Laboratory, France
Yulei Wu, University of Exeter, UK

PESARO Publicity Chairs
José Miguel Jiménez, Universitat Politecnica de Valencia, Spain
Francisco Javier Díaz Blasco, Universitat Politècnica de València, Spain
Ali Ahmad, Universitat Politècnica de València, Spain

Laura Garcia, Universidad Politécnica de Cartagena, Spain
Sandra Viciano Tudela, Universitat Politecnica de Valencia, Spain

PESARO 2025

Committee

PESARO Steering Committee

Mohammad Rajabali Nejad, University of Twente, the Netherlands
Rémy Houssin, Université de Strasbourg - ICube Laboratory, France
Yulei Wu, University of Exeter, UK
Wolfgang Leister, Norsk Regnesentral, Norway

PESARO 2025 Publicity Chairs

José Miguel Jiménez, Universitat Politecnica de Valencia, Spain
Francisco Javier Díaz Blasco, Universitat Politècnica de València, Spain
Ali Ahmad, Universitat Politècnica de València, Spain
Laura Garcia, Universidad Politécnica de Cartagena, Spain
Sandra Viciano Tudela, Universitat Politecnica de Valencia, Spain

PESARO 2025 Technical Program Committee

Mohammad AlMasri, Nvidia, USA
Ehsan Atoofian, Lakehead University, Canada
Kaustav Basu, Arizona State University, USA
Morteza Biglari-Abhari, University of Auckland, New Zealand
Chérifa Boucetta, University of Reims Champagne Ardenne, France
Lelio Campanile, University of Campania Luigi Vanvitelli, Italy
Pasquale Cantiello, University of Campania Luigi Vanvitelli, Italy
Frank Coolen, Durham University, UK
Faten Fakhfakh, National School of Engineering of Sfax, Tunisia
Victor Flores, Universidad Católica del Norte, Chile
Rita Girao-Silva, University of Coimbra & INESC-Coimbra, Portugal
Marco Gribaudo, Politecnico di Milano, Italy
Christoph-Alexander Holst, inIT - Institute Industrial IT, Germany
Rémy Houssin, Université de Strasbourg - ICube Laboratory, France
Benoit Iung, Lorraine University, France
Christos Kalloniatis, University of the Aegean, Greece
Atsushi Kanai, Hosei University, Japan
Liuwang Kang, University of Virginia, USA
Sokratis K. Katsikas, Norwegian University of Science and Technology, Norway
Georgios Keramidas, Think Silicon S.A., Greece
Michel A. Kinsy, STAM Center | Arizona State University, USA
Vincent Latzko, Technische Universität Dresden, Germany
Wolfgang Leister, Norsk Regnesentral, Norway
Michele Mastroianni, University of Campania -Luigi Vanvitelli, Italy
Ilaria Matteucci, IIT-CNR, Italy

Mohamed Nidhal Mejri, Paris 13 University, France
Weizhi Meng, Lancaster University, UK
Zewei Mo, University of Pittsburgh, USA
Andrey Morozov, University of Stuttgart | Institute of Industrial Automation and Software Engineering (IAS), Germany
Mohammad Rajabali Nejad, University of Twente, the Netherlands
Mohamed Nounou, Texas A&M University at Qatar, Qatar
Tuan Phung-Duc, University of Tsukuba, Japan
Vladimir Podolskiy, Technical University of Munich, Germany
Omar Smadi, Iowa State University, USA
Kumiko Tadano, NEC Corporation, Japan
Eirini Eleni Tsiropoulou, Arizona State University, USA
Alexandre Voisin, Université de Lorraine, France
Yulei Wu, University of Exeter, UK
Patrick M. Yomsi, CISTER Research Unit - ISEP/IPP, Portugal
Bingyi Zhang, University of Southern California, USA
Piotr Zwierzykowski, Poznan University of Technology, Poland

Copyright Information

For your reference, this is the text governing the copyright release for material published by IARIA.

The copyright release is a transfer of publication rights, which allows IARIA and its partners to drive the dissemination of the published material. This allows IARIA to give articles increased visibility via distribution, inclusion in libraries, and arrangements for submission to indexes.

I, the undersigned, declare that the article is original, and that I represent the authors of this article in the copyright release matters. If this work has been done as work-for-hire, I have obtained all necessary clearances to execute a copyright release. I hereby irrevocably transfer exclusive copyright for this material to IARIA. I give IARIA permission or reproduce the work in any media format such as, but not limited to, print, digital, or electronic. I give IARIA permission to distribute the materials without restriction to any institutions or individuals. I give IARIA permission to submit the work for inclusion in article repositories as IARIA sees fit.

I, the undersigned, declare that to the best of my knowledge, the article is does not contain libelous or otherwise unlawful contents or invading the right of privacy or infringing on a proprietary right.

Following the copyright release, any circulated version of the article must bear the copyright notice and any header and footer information that IARIA applies to the published article.

IARIA grants royalty-free permission to the authors to disseminate the work, under the above provisions, for any academic, commercial, or industrial use. IARIA grants royalty-free permission to any individuals or institutions to make the article available electronically, online, or in print.

IARIA acknowledges that rights to any algorithm, process, procedure, apparatus, or articles of manufacture remain with the authors and their employers.

I, the undersigned, understand that IARIA will not be liable, in contract, tort (including, without limitation, negligence), pre-contract or other representations (other than fraudulent misrepresentations) or otherwise in connection with the publication of my work.

Exception to the above is made for work-for-hire performed while employed by the government. In that case, copyright to the material remains with the said government. The rightful owners (authors and government entity) grant unlimited and unrestricted permission to IARIA, IARIA's contractors, and IARIA's partners to further distribute the work.

Table of Contents

A Systems Approach to Modelling Safe Behaviour of Maritime Control Systems Using the Composition, Environment, Structure, and Mechanisms (CESM) Metamodel

Odd Ivar Haugen
Group Research and Development department, DNV AS
Trondheim, NORWAY
e-mail: odd.ivar.haugen@dnv.com

***Abstract*—Society increasingly relies on complex systems whose behaviour is determined, not by the properties of each part, but by the interaction between them. The behaviour of such systems is emergent. Modelling emergent system behaviour requires a systems approach that incorporates the necessary concepts that are capable of determining such behaviour. The CESM metamodel (Composition, Environment, Structure, Mechanisms) is a model of system models. A set of system models needs to address the elements of CESM at different levels of abstraction to be able to model the behaviour of a complex system. Modern ships contain numerous sophisticated equipment, often accompanied by a local safety system to protect their integrity. These control systems are then connected into a larger integrated system in order to achieve the ship's objective or mission. The integrated system becomes, what is commonly known as, a system of systems which can be termed a complex system. Examples of such complex systems are the ship's dynamic positioning system and the power management system. Three ship accidents are provided as examples of how system complexity may contribute to accidents. Then, the three accidents are discussed in terms of how the Multi-Level/Multi-Model Safety Analysis might catch scenarios such as those leading to the accidents described.**

Keywords-emergent properties; cesm metamodel; multi-level/multi-model safety analysis; safety; system complexity; systems approach.

I. Introduction

The number of ship control systems has increased tremendously in the last 25 years. There are dedicated control systems related to power generation, such as controlling switchboard circuit breakers, stopping and starting generators, and reducing load to avoid blackouts. Moreover, fire and gas systems may start deluge systems, leading to the automatic stop of power generation equipment. Dynamic Position Systems (DPS) rely upon the fact that there is adequate thrust available to maintain position. Local dedicated thruster control systems control pitch (Angle of the thruster blade) and the RPM (Revolutions Per Minute of the thruster blade) of the thruster, which is part of the DPS. There are automatic shut-down systems whose sole purpose is to protect the equipment.

A more and more prevailing challenge is to gain oversight over how the control systems interact and how an action taken by one local control/safety system affects other control systems. The control system can be seen as located in a hierarchy of control at different levels of authority and responsibility. An action taken by one local safety system may inadvertently shut down equipment necessary for another control system to work as intended.

Lately, there have been accidents outside the Norwegian coast where, at least one of them, in a worst-case scenario, could develop into the worst maritime catastrophe in modern history, on par with the sinking of Titanic.

Today, methods for safety analysis and assurance of maritime systems have not kept up to the task of dealing with increased system complexity due to increased tight integration between the control systems.

This paper suggests a framework for system analysis to adequately deal with increased system complexity, using maritime control systems and maritime accidents as a background and examples.

The remainder of this paper is structured as follows. Section (II) "A few recent maritime accidents on the Norwegian Coast" briefly describe three recent maritime accidents on the Norwegian coast, which serve to motivate our discussion on increased system complexity. In Section (III) "Commonalities between the accidents", we examine the common factors among these accidents, highlighting the role of complex interactions among control and safety systems. Section (IV) "Reductionism versus systems thinking" discusses reductionism and explains why it is insufficient for ensuring safety in modern maritime systems. Section (V) "The Systems Approach to handle system complexity and emergence" introduces the CESM metamodel and explains how its four elements—Composition, Environment, Structure, and Mechanisms—offer a systemic view of system-level behaviour. Section (VI) "System analysis" outlines a systematic analysis approach based on Multi-Level, Multi-Model Safety Analysis (ML/MM-SA). In Section (VII) "Application of the method", we demonstrate how this method might have captured the accident scenarios described earlier. Finally, Section (VIII) "Conclusion" concludes and provides avenues for future work.

II. A few recent maritime accidents on the Norwegian Coast

To set the stage, this section will go through three recent accidents on the Norwegian Coast. The actual loss in each accident differs, and the amount of information from the accident investigation also varies. The motivation is not to question the official stated direct cause of the accident. Indeed, the cause of one of the accidents is not known. Instead, we use these accidents to argue that system complexity could have been a contributing factor, even if this is not explicitly mentioned in the accident reports.

 ISBN: 978-1-68558-280-7

A. Vessel: Sjoborg

Sjoborg is a supply vessel that, at the time of the accident, operated as a Platform Supply Vessel (PSV) for the Norwegian energy company Equinor. Equinor is the operator of the Statfjord oil field, where one of the production platforms is Statfjord A [1]. Statfjord A is the world's largest Condeep (CONcrete DEEP water structure) production platform [2], with a weight of 290.000 tonnes and a storage capacity of 1.3 million barrels of oil. It is a fixed platform with a total height of 270 metres, standing on three concrete legs.

Sjoborg's power system is a hybrid [3], that is, a combination of diesel generators and battery power. This design introduces additional control systems related to the battery system compared to a more traditional system that is based exclusively on diesel generators.

A PSV carries goods and equipment to and from the platforms. In general, when such vessels load or discharge goods, they need to be stationary at a particular position in relation to the platform due to, for instance, the sea and weather conditions or the location of the cranes onboard the platform.

For a floating vessel to maintain its position, it uses Dynamic Positioning (DP). Simplified, a DP system maintains a fixed vessel position by providing thrust to counteract the environmental forces.

A vessel such as Sjoborg that operates on DP close to an offshore oil installation will typically be classified in accordance with the DP guideline published by the International Maritime Organization (IMO) - Maritime Safety Committee (MSC): IMO MSC.1/Circ.645 - Equipment class 2 [4]. For such vessels, loss of position shall not occur in the event of a single failure in any active component or system.

On 7 June 2019, while loading/discharging alongside Statfjord A, the control systems onboard Sjoborg initiated a power reduction in response to an event. This automatic power reduction resulted in a series of events that eventually resulted in the Sjoborg colliding with the platform [5].

The initial event led to a communication network failure in the blackout safety system; this led to the main switchboard frequency measurement being lost (the frequency was at this point not affected), which again led to the activation of the load reduction mode, which led to that all the power from all thrusters where reduced to 10%-15% of their maximum output, this led to a discrepancy between the DP systems's thruster RPM command signal and the feedback from the thrusters, which eventually led to that two thrusters where automatically shut down.

In the end, Sjoborg did not have enough power to counteract the environmental forces, drifted towards the platform, and eventually collided.

One of the Sjoborg crew was hit in the face by a diesel hose; fortunately, it did not result in a fatality, but under slightly different circumstances, it could have. Moreover, Sjoborg suffered material damage, and the lifeboats onboard Statfjord A were damaged. This led to the helicopter evacuation of 218 people from Statfjord A.

We see here that the analysis of the behaviour of the system did not capture how the different control systems interacted as a response to the initial event.

B. Vessel: MS Richard With

MS Richard With is one of the "Hurtigruten" vessels trafficking the Norwegian coastline, and it has a capacity of 590 people [6]. In 2022, the power system onboard MS Richard With was converted to a hybrid power system, that is, diesel generators and battery package [7].

On the sea trial, before going into ordinary operation, the ship grounded caused by a blackout in the power system [8]. As there was no public accident investigation, it is difficult to get information about the direct cause. The only known cause was "technical system failure" [9].

Luckily, the accident happened before the ship went into regular traffic along the Norwegian coast. The grounding only caused damage to the ship.

Although "technical system failure" does not provide much insight into what actually caused the blackout, it is worth noticing that this ship was also rebuilt to hybrid power, resulting in a number of additional control systems, just as on Sjoborg. We stress that we do not know the cause of this accident, so we do not conclude that the accident was related to increased complexity as a result of hybrid power. However, a hybrid power system will, in general, increase the complexity of the power system.

C. Vessel: Viking Sky

Viking Sky is a cruise ship equipped for 930 passengers [10]. The cruise ship is classified in accordance with IMO MSC.216, which includes the "Regulation 21 Casualty threshold, safe return to port and safe areas" [11]. Safe Return to Port (SRtP) requires that a vessel be able to return to port under its own propulsion after a casualty that does not exceed a certain threshold.

On the afternoon of 23 March 2019, with a total of 1374 people onboard, the ship experienced a total blackout and lost all propulsion while crossing Hustadvika at the coast of Norway during a heavy storm. The ship was pushed or drifted towards the reefs in Hustadvika. Hustadvika is a well-known area with difficult sailing conditions [12].

The direct cause of the blackout was a combination of low oil levels in all engine lubrication oil tanks and heavy rolling and pitching, causing the hose that is supposed to suck lubrication oil from the tanks to the engine to instead suck air, which again caused the lubrication oil pressure to drop, resulting that the engine safety system kicked in and stopped all engines [13]. The purpose of the engine safety system is to protect the engine against damage.

It was estimated by the accident report that the ship was about one ship length from the reefs when the crew managed to restart two of the engines after 39 minutes so that power was restored and they could get clear of the reefs. If the crew did not get to restart the engines in time, this could have developed into the worst maritime catastrophe in modern times [13]. In

the Titanic catastrophe, about 1500 people died [14], and in the fire and sinking of MS Estonia, 852 people died [15]. The Viking Sky accident could have caused as many people's lives as those two catastrophes.

IMO MSC.216 is not as strict as IMO MSC.1/Circ.645 for DP Equipment class 2, so there is no requirement that a blackout cannot occur. However, it requires that the power be restored in due time to avoid situations like this one, and it is based on the same fundamental principles such as redundancy and component reliability.

There is no discussion about the direct cause of the accident; that is indisputable. However, in the context of the discussion about the increased system complexity, and the many control systems, one could start to ask why the crew could not start the engines quicker than after more than 30 minutes when the situation was so critical. Indeed, in an interview with the Norwegian National Broadcaster, NRK, the pilot stated (translated to English from Norwegian): "I really missed a button that said override on it" [16].

This opens a number of (rhetorical) questions like: "Why can an engine safety controller be allowed to stop all engines at the same time and no one can prevent it"?, and "Who has the best oversight over the situation? The engine safety system, or the pilot on the bridge"?, and "What is more valuable? A couple of diesel engines, or 1374 human lives"?

These questions point to some interesting discussions, not only about the oil level in the tank, but also about what controller should have the highest authority, the human controller on the bridge, or a safety system whose sole purpose is to protect equipment. There might, of course, be good reasons for the design, and we are not going to provide design advice, but the question remains: who should control the "override button"?, and, should there even be an "override button"?

III. Commonalities between the accidents

All ships had redundant equipment and were certified in accordance with relevant IMO safety guidelines. In all cases, neither the equipment redundancy, nor the equipment reliability prevented the accidents from occurring. Both Sjoborg and MS Richard With, utilise a hybrid power system, which is known to create increased system complexity due to extra control and safety systems. These control systems need to interact in such a way that safety is maintained. In the case of Sjoborg and Viking Sky, a safety system completely defeated the redundancy philosophy.

The commonality between all accidents may be said to be a lack of understanding of the behaviour of the integrated control system, including the actions taken by different control systems and their associated authority. A reservation needs to be made for MS Richard With because of a lack of information.

IV. Reductionism versus systems thinking

The IMO system safety standards are based on redundancy and equipment reliability, that is, reductionism. Reductionism interprets the world as a pile of things [17] so the world can be understood by investigating these parts. This leads to system safety becoming a question about avoiding component and equipment failures by highly reliable components and/or the concept of component redundancy. However, as we saw in the previous examples, redundancy is not the "silver bullet" to safety in complex systems.

This view on safety is typically represented by using the method Failure Mode and Effect Analysis (FMEA) for safety analysis [4][11][18]. FMEA was invented as a reliability analysis, not a safety analysis [19][20]. Charles O. Miller, one of the founders of system safety, put this clearly: "distinguishing hazards from failures is implicit in understanding the difference between safety and reliability" [21].

While reductionism may have served some industries well in the past, where the safety principle is founded upon a dedicated safety function where there is one single action that brings the system into a predetermined single safe system state. This safety function is achieved by a controller that has the highest authority. Such systems are characterised as KISS (Keep It Simple, Stupid). These safety systems are found in the process industry, such as oil production. If a process gets too hot, or too high pressure, or some flow is too high or too low, the actions would often be to open or close a valve, or, by some means, shut down the process or flow. Typically, these actions can be summarised as removing energy from the system, which would bring the system into its single predefined safe state. The reliability of the components in the safety system may determine safety in such simple systems.

Recall what happened in the case of Sjoborg when the available power was removed from all thrusters, followed by a shutdown of thrusters 1 and 3. This action of removing energy may have brought the switchboard into its "safe" state, but it definitely did not bring the ship into a safe state. The same explanation can be applied to Viking Sky; the engine safety system brought the engine into its "safe" state; however, the lack of power to the ship propulsion system resulted in one ship length from potentially the worst maritime catastrophe in modern times.

The complexity of many of today's industrial safety-critical systems, including ship systems, requires a shift in how we understand safety. Safety is an emergent property [21] that cannot be fully understood through reductionism because the property of interest is not a property of the components but of the system.

This complexity applies not only to the system of interest but also to the environment in which it is operating. As with the system, neither can its environment be seen as a "pile of things", but as a set of complex systems. This definitely applies to an autonomous ship sailing in a shipping lane where the object detection system of the autonomous navigation and collision avoidance system cannot only detect other ships as objects or "things" in its environment, but must also understand their intended route, their manoeuvring capabilities, in general, the system state of this "thing" called a ship which must be expanded to more than of its physical appearance.

V. The Systems Approach to handle system complexity and emergence

Complex socio-technical systems consist of components and agents (human and artificial) that interact and perform a series of interdependent actions to achieve goals in different environments that, themselves, are systems with non-trivial interacting components. The system properties and behaviour cannot be understood by investigating single components inside the system. Due to the interaction of interdependencies between components and agents, and between the system and its environment, the system properties and behaviour are emergent.

Such properties do not exist in each component, but emerge due to their interactions. By reducing the system into its components, the properties are lost, and therefore become unobservable. Such properties can be said to be computationally irreducible [22].

The growth/decline of macroeconomics and the stock market, the social life of army ants, the wetness of a raindrop, human culture, the global climate, a city's resilience against a catastrophe, and system safety are all examples of emergent behaviour or properties [21][23][24].

A. Complexity - emergence and the CESM metamodel

Complexity and emergent behaviour, or emergent properties, are closely related; hence, the science of emergence is really about complexity [23][25]. There is no single and all-encompassing definition of either complexity or emergence. In the same way, a universal understanding of how to measure them does not exist among either scientists or philosophers [24]. One reason for the lack of a definition is that complexity can come in many forms, such as [21][24][26]:

- Size,
- level of entropy,
- logical and functional depth,
- level and amount of interaction and interdependencies among system entities,
- non-linear causes and effects,
- feedback loops,
- number of system states,
- intricate transition rules between states.

The forms of complexity indicate intractability, non-trivial ways of understanding, explaining and predicting the behaviour of a complex system.

However, it is important to notice that complexity and emergence are properties of the system, not of epistemology [27]. Explained emergence (and complexity) is still emergence [28]; that is, a system does not cease to be complex just because we understand (to a certain degree) its behaviour.

A way to understand and analyse complex systems and emergence is to model the system behaviour in terms of its composition, structure, mechanisms and the environment in which it operates. These system aspects are termed the CESM metamodel [28]:

- **Composition** (C): Collection of all the parts or objects in the system.
- **Environment** (E): Systems outside (excluded from) the target system, but act upon, or are acted upon by, the target system.
- **Structure** (S): The relationships and bonds among the system agents and between the system agents and the environment.
- **Mechanisms** (M): The processes that make the system behave in the way that it does.

The emergent behaviour becomes a function of the above elements; that is, any system s may be modelled, at any given instance, as the quadruple: $\mu(s) = <C(s), E(s), S(s), M(s)>$.

Complexity can be understood in the context [29]:

- **Composition**: Number of system objects, parts and elements. Size of composition hierarchies.
- **Environment**: Size of state space, number of agents and their autonomy, (lack of) rules of interaction with the system.
- **Structure**: The stability of the relationship, responsibility and authority between the system agents, and between the system agents and the environment. The degree of cooperation needed to achieve a goal.
- **Mechanisms**: Number of functions, what agent can/must perform them, needed resources, number of preconditions, possible postconditions, and the control of their execution.

In short, emergent properties result from the conceptual interaction between the elements in the CESM metamodel [30], and complexity can be thought of by how intricate these interactions are.

To investigate the nature of such interactions, the bonds, roles, and responsibilities of agents (the Structure), how they interact (the Mechanisms), and the properties of the system components (the Composition) must be analysed and synthesised.

The above pseudo-definition of emergence and complexity does not entirely describe what these concepts entail; however, it is helpful when developing a framework for understanding and analysing complex systems.

B. Levelism

What constitutes a system depends on the observer's point of view [31]. For two different observers, the same entity may be seen as a system with interacting components, and for another, it can be seen as a (single) component within a larger system.

System behaviour can be analysed (explained) at different Levels of Abstractions (LoAs), depending on the observer's viewpoint; that is, depending on the knowledge we seek [32]. Hence, interactions and dependencies must also be explained at different LoAs. This means that (abstract) system constituents (items, agents and actions) must be identified at different LoAs [30].

An analysis at one LoA is not "better" than at another; they are just different because they provide different kinds of knowledge about the system. The search for knowledge in the current context, driven by the objective of the analysis, guides our choice for LoAs, that is, epistemic levelism.

We may divide levelism (LoA) into epistemic and ontological. Epistemic levelism addresses the kind of knowledge that we seek; ontological levelism is how (we choose to) divide the system into levels of detail. The two kinds of levelism are

often closely related; that is, how the system is divided into levels is often related to what kind of knowledge we seek.

C. System models

Models representing the system are abstractions of constituents and their relationships and bonds. The entities within the system models are also abstractions. The entities included in a system model at certain LoAs may not exist in the actual system or even be planned to exist. The names of the system model entities may indicate their function, role, type, or other features.

More than a single system model is needed to address μ(s). As the conceptual interaction between the elements of the CESM metamodel is both necessary and sufficient to describe any system behaviour, the collection of system models must address every element of the CESM metamodel at the LoAs (epistemic and ontological) needed to gain adequate knowledge [29]. Moreover, they must also be connected so that the emergent system behaviour, μ(s), becomes observable.

For each element in the CESM metamodel, we can assign different model categories. Moreover, the model categories must be connected to elicit μ(s). The following model categories represent the CESM metamodel:

- Composition: **Object model** representing the system elements and components and their ontological relationship to each other.
- Environment: Also modelled as a system containing all aspects of the CESM metamodel, which means that the environment must be represented by models representing the composition, structure and mechanisms (our target system is part of the environment of its environment).
- Structure: **Agent model** includes entities such as controllers, actuators, sensors, humans, and AI subsystems. The agent concept includes authority, responsibility, goals, concerns, motivation, and wishes (humans).
- Mechanisms: **Function model** represents the operations that must be performed (by the agents) to achieve goals.

A specific system model is an instantiation of the above categories. A control structure including a controller, control actions, feedback, and a controlled process known from Systems-Theoretic Process Analysis (STPA) [21] is one instance of an agent model. Another agent model may focus more on the agent's goals, motivation, concerns and wishes, like a model used in a stakeholder analysis where social and business aspects are emphasised.

A function model may focus on the preconditions, resources, and timing for achieving it, like the model used in the Functional Resonance Analysis Method (FRAM) [26]. Or, it may focus on functional dependencies to other functions, like in the Functional Analysis System Technique (FAST) [33].

The different models give different views of the same system, which means that the models should be consistent. Every model has qualities the others lack; however, they need points of contact to ensure their consistency; they need to "borrow" some aspects from each other [29]. The models should be distinguished, not detached or isolated. On top of these borrowed aspects, consistency rules regulate their relationship.

These relationships and rules increase rigour (formalism) in revealing the system behaviour. This is important for the objectivity, the transparency, and thereby the trustworthiness in any context in which these models are used.

Such consistency rules and relationships among multiple system models naturally lend themselves to a Model-Based Safety Analysis (MBSA) toolchain, which is itself an application of Model-Based Systems Engineering (MBSE) as advocated by the International Council on Systems Engineering (INCOSE) community. In this sense, frameworks like SysML can capture and integrate the four CESM elements (Composition, Environment, Structure, and Mechanisms) into a single authoritative system model (Figure 1). Only the model categories for the system are included, not for the environment; however, these model categories should also be incorporated into the environment's system model.

By grounding the safety analysis in an MBSE environment, one follows established INCOSE guidelines for improving system complexity management via formal modelling and consistent architectures. In an MBSA setting, the different views introduced here (object, agent, function, and environment models) are instantiated in SysML, enabling partially or fully automated generation of safety analysis artefacts. Consequently, emergent properties, dynamic behaviours, and critical interdependencies become explicit model elements. This helps ensure rigour (formalism), transparency, and trustworthiness in how complex maritime control systems are designed, analysed, and assured.

From the above discussion, we can conclude that the method for analysing complex systems must be conducted using multiple models at multiple levels of abstraction. The method is called Multi-Level, Multi-Model Safety Analysis (ML/MM-SA) [34].

VI. System analysis

A system may fail to meet expectations due to defects in the elements, or in a combination of the elements, in the CESM-model [35], [30]:

- **Composition**: E.g., missing components, inappropriate component types, component redundancy, etc.
- **Environment**: The system works outside the operational environment for which it was designed.
- **Structure**: E.g., inappropriate or lack of connections, bonds, relationships, or associations between the components.
- **Mechanisms**: E.g., inappropriate or missing rules of interaction between the components

For any analytical method or simulation model, it is important to know the extent to which it explores these defect causes, or combinations thereof, including their potential evolution over time. It indicates causes rooted in the system design, and that should, therefore, be mitigated in the system design phase. The list below, on the other hand, indicates the context in which the system may fail to meet expectations:

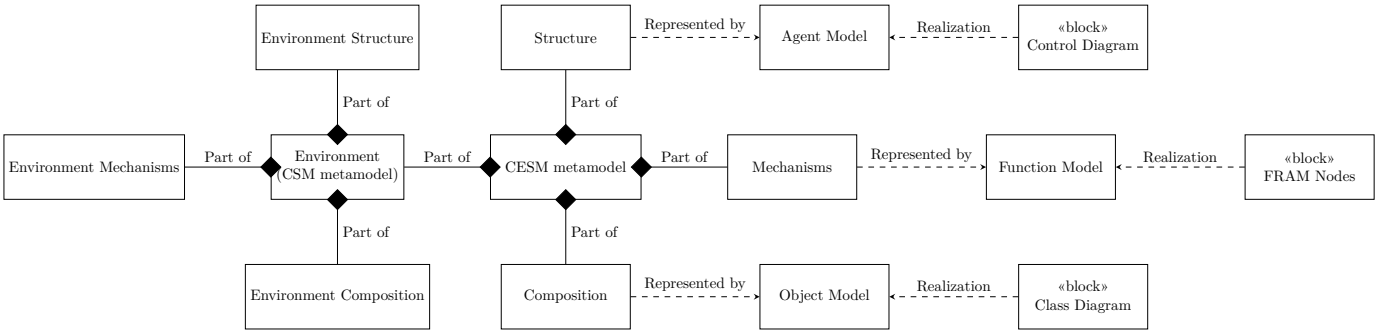

Figure 1. SysML Block Definition Diagram (BDD) of the CESM metamodel.

- **Composition**: The current state of the components, like fully operational /overloaded /degraded /stopped /failure mode, etc.
- **Environment**: State space of the environment, such as temperature /daylight /humidity of a physical environment, or, for an environment consisting of other agents, their speed /course /location, or their (presumed) intention /operational mode etc.
- **Structure**: The state of the relationship between the system components or the environment. This may depend on the current role of a component in relation to other components, or towards the environment, or the current operational mode of the system.
- **Mechanisms**: The current rule-set of interaction; this may depend upon the state of the structure (e.g., roles and relationships), but also on the current state of the composition (e.g., a component, like a sensor, may be out of service and thereby other rule-sets are active).

By combining the items in the above two lists, it becomes clear that there is a substantial number of ways leading to what is known as the combinatorial explosion, in which a system may fail to meet expectations. Therefore, building confidence that a system will meet expectations in all possible situations can be highly challenging. Nevertheless, a system safety analysis must encompass both the above lists in a systematic way.

VII. Potential Application of the Method

In this section, we illustrate how the proposed method could have helped identify or predict the scenarios leading to the accidents described in Section II. Rather than presenting a complete, real-world application or post-accident analysis, we provide a hypothetical demonstration of how the method's core concepts—examining Composition, Environment, Structure, and Mechanisms—might uncover unsafe control actions and emergent behaviours. This illustrative approach highlights the potential of the method to capture the behaviour of complex systems. Still, it does not constitute a full validation of the method against actual case-study data.

The method has also been used as the basis for making DNV-specific guidelines for the DP industry and for guidelines for the assurance of Artificial Intelligence (AI). Moreover, the method has also been used to analyse a subsea Christmas tree.

In the case of Sjoborg, the different local control systems, including the safety systems, such as the blackout prevention system, did not properly interact in such a way that an adequate amount of power was maintained for station keeping using the DP system.

An agent model, such as the control structure found in STPA, represents the system structure ("S" in CESM) that investigates the connections, and relationships between the controllers in the different subsystems and of the safety systems. Such an investigation would focus on the authority, responsibility, and goals (purpose) of each controller associated with the DP system. Moreover, the control structure in STPA also includes the concept of control actions, which is a function achieved by a controller. STPA also specify a set of guidewords to identify unsafe control actions. In particular, STPA identifies how control actions could become unsafe if they are provided too early, too late, or not at all when needed. By examining these possible deviations, STPA makes the system's pathways to hazard more transparent. This is the way by which a controller, through its control action, may or may not set the controlled process into a hazardous state. Although the STPA guidewords are a good help in identifying unsafe control actions, a more explicit approach is to develop a function model ("M" in CESM), such as the one used in FRAM, to systematically investigate the timing, resources, and other conditions that might either hinder a safe action from being achieved, or promote an unsafe action to be achieved by the controller. Such scenarios may be caused by an abnormal state of a system component ("C" in CESM). It is important to notice that the state of the elements in the composition need not be in a failure state to affect how a controller achieves a function. From the investigation report of the accident with Sjoborg, it was indicated that it was a component failure that initiated the scenario, however, still, a vessel like Sjoborg should be able to maintain its position despite such a failure.

In the case of Viking Sky, the method would address the fact that the engine safety system possesses maximum authority over the shutdown of the diesel engines. Whether this design would be maintained after identifying this fact, or the crew would get access to an override button, would, of course, be up to the design team and the class society responsible for approving the design. It is worth noting that the human operators are treated

as controllers in the same manner as the automatic controllers, therefore, the authority and responsibility of the pilot onboard Viking Sky would be taken into account in the analysis.

In the case of MS Richard With, it is difficult to say in detail whether this method would identify the scenario leading to the blackout during the sea trial because of lack of information; however, given the power system was hybrid, it may not be surprising that this method also would shed light in this case.

DNV, Equinor and Shell initiated a Joint Development Project (JDP) together with the DP industry to address the underlying cause of the Sjoborg accident. This project resulted in a Recommended Practice (RP): DNV-RP-0684 "Dynamic Positioning Systems – systems integration" [36], which is a guideline to be used by the industry to be able to analyse and predict such scenarios causing the Sjoborg accident. This is a bespoke guideline for the DP industry, but the theoretical foundation is the method described in this paper.

Safety-related control systems based on AI are being deployed. In the maritime domain, autonomous navigation has already been deployed in several places in Europe. DNV has made a Recommended Practice: DNV-RP-0671 "Assurance of AI-enabled Systems" [37] that requires that the AI system is modelled in accordance with the CESM-metamodel at all relevant abstraction levels as per described in this paper.

DNV, together with the subsea oil and gas industry operating on the Norwegian continental shelf, created a Joint Industry Project (JIP) to address increased system complexity in safety-critical subsea systems. In this project, an analysis of a subsea Christmas tree was performed using this method [38].

VIII. Conclusion

Ship accidents occur can be related to increased ship system complexity. Methods for analysing the behaviour of such systems are based on a reductionist view of the world, which sees it as a "pile of things". Therefore, such methodologies are conceptually inadequate to achieve the objective of such analysis, and the practitioners end up looking for the needle in the haystack.

This paper has described an alternative methodology based on systems thinking. This method acknowledges that the behaviour of complex systems is emergent. Such properties emerge as a result of the interaction and interdependencies within the system constituents, and between the system and the environment in which it operates. One such property is system safety.

These principles have already led to practical outcomes. For instance, the method directly informed the development of DNV's recommended practices for dynamic positioning systems and AI assurance, and it was used in the analysis of a subsea Christmas tree.

By explicitly addressing composition, environment, structure, and mechanisms at multiple levels of abstraction, this approach advances the literature on maritime safety analysis and provides a concrete, systems-based framework for tackling emerging technological challenges.

References

[1] Equinor, 'The Statfjord area', Accessed: 26th Mar. 2025. [Online]. Available: https://www.equinor.com/energy/statfjord.

[2] *Facts about Statfjord "A":The World's Largest Condeep Production Platform*. Aker Group, 1979, ISBN: nb.bibsys.no (991000918904702202).

[3] 'Fact Sheet – Sjoborg', Skansi Offshore, 2012, Accessed: 18th Mar. 2025. [Online]. Available: https://skansi.fo/fleet/sjoborg/.

[4] *Guidelines for vessels with dynamic positioning systems (MSC Circular 645)*, Guideline, Obsolete, replaced by MSC.1/Circ.1580 on 9~June 2017}.

[5] A. Oplenskedal, L. G. Bjørheim and R. L. Leonhardsen, 'Investigation of collision between Sjoborg supply ship and Statfjord A on 7 June 2019', Statens havarikommisjon, Investigation report 001037045, Jun. 2019.

[6] 'MS Richard With Itinerary, Current Position, Ship Review', CruiseMapper, Accessed: 26th Mar. 2025. [Online]. Available: https://www.cruisemapper.com/ships/MS-Richard-With-772.

[7] 'Hurtigrutens første grønne hybridskip klar for seilas langs norskekysten (The first green hybrid ship from Hurtigruten ready to set sail along the cost of Norway)', *Maritimt Magasin*, 23rd Sep. 2022.

[8] C. Salas-Gulliksen, 'Hurtigruten har grunnstøtt nord for Sognefjorden (Hurtigruten has grounded north of Sognefjorden)', NRK, 5th Aug. 2022, Accessed: 26th Mar. 2025. [Online]. Available: https://www.nrk.no/vestland/hurtigruten-har-grunnstott-nord-for-sognefjorden-1.16058532.

[9] T. Stensvold, 'Hurtigruten: Teknisk feil var årsak til grunnstøtingen (Hurtigruten: The cause was a technical failure)', *Tu.no*, 5th Aug. 2022.

[10] 'Viking Sky', Accessed: 26th Mar. 2025. [Online]. Available: https://www.vikingcruises.co.uk/oceans/ships/viking-sky.html.

[11] *Resolution MSC.216(82) - Adoption of Amendments to the International Convention for the Safety of Life at Sea, 1974 as Amended - (Adopted on 8 December 2006)*, Guideline, Dec. 2006.

[12] M. K. Korsnes, 'Difor er det så dramatisk å få motorstopp over Hustadvika (This is why an engine stop gets so dramatic across Hustadvika)', *NRK, dk*, 26th Aug. 2021.

[13] NSIA, 'Loss of propulsion and near grounding of Viking Sky, Hustadvika, Norway, 23 March 2019', Norwegian Safety Investigation Authority, Accident investigation MARINE 2024/05, Mar. 2025.

[14] P. K. Sebak, *Titanic*, in *Store norske leksikon*, 18th Jun. 2024.

[15] P. Sebak, *M/S Estonia*, in *Store norske leksikon*, 20th Jun. 2024.

[16] O. Bjørneset, 'Losen på «Viking Sky» meiner automatikken må kunne overstyrast (The pilot onboard the "Viking Sky" suggests that it should be possible to override the automatic control)', *NRK, dk*, 24th Sep. 2019.

[17] M. Bunge, *Emergence and Convergence: Qualitative Novelty and the Unity of Knowledge*, Reprint edition. Toronto: University of Toronto Press, Scholarly Publishing Division, 9th Jul. 2014, 344 pp., ISBN: 978-1-4426-2821-2.

[18] *MSC.1/Circular.1580 – Guidelines for Vessels and Units with Dynamic Positioning (DP) Systems – (16 June 2017)*, Guideline, Jun. 2017, Current.

[19] N. Leveson, *SafeWare: System Safety and Computers*. Reading, Mass: Addison-Wesley, 1995, 680 pp., ISBN: 978-0-201-11972-5.

[20] C. A. Ericson, 'Failure Mode and Effects Analysis', in *Hazard Analysis Techniques for System Safety*, John Wiley & Sons, Ltd, 2005, pp. 235–259, ISBN: 978-0-471-73942-5. DOI: 10.1002/0471739421.ch13.

[21] N. G. Leveson, *Engineering a Safer World: Systems Thinking Applied to Safety*. Cambridge, Massachusetts: MIT Press, 13th Jan. 2012. DOI: 10.7551/mitpress/8179.001.0001.

[22] S. Wolfram, 'Undecidability and intractability in Theoretical Physics', in *Emergence*, Cambridge, Massachusetts: The MIT Press, 2008, pp. 387–393, ISBN: 987-0-262-02621-5.

[23] J. H. Holland, *Complexity: A Very Short Introduction* (Very Short Introductions 392), First edition. Oxford, United Kingdom: Oxford University Press, 2014, 95 pp., ISBN: 978-0-19-966254-8.

[24] M. Mitchell, *Complexity: A Guided Tour*. New York, NY: Oxford University Press, 2011, 349 pp., ISBN: 978-0-19-979810-0.

[25] M. M. Waldrop, *Complexity: The Emerging Science at the Edge of Order and Chaos* (A Touchstone Book), 1. Touchstone ed. New York, NY: Touchstone, 1993, 380 pp., ISBN: 1-5040-5914-X 978-1-5040-5914-5.

[26] E. Hollnagel, *FRAM: The Functional Resonance Analysis Method, Modelling Complex Socio-Technical Systems*. Ashgate Publishing Limited, 2012.

[27] M. A. Bedau, 'Is Weak Emergence Just in the Mind?', *Minds and Machines*, vol. 18, no. 4, pp. 443–459, 1st Dec. 2008, ISSN: 1572-8641. DOI: 10.1007/s11023-008-9122-6.

[28] M. Bunge, *Emergence and Convergence: Qualitative Novelty and the Unity of Knowledge* (Toronto Studies in Philosophy). Toronto ; Buffalo: University of Toronto Press, 2003, 330 pp., ISBN: 978-0-8020-8860-4.

[29] O. I. Haugen, 'The Systems Approach', in *Demonstrating Safety of Software-Dependent Systems; With Examples from Subsea Electric Technology*, T. Myhrvold and M. van der Meulen, Eds., DNV AS, 2022, pp. 145–163, ISBN: 978-82-515-0324-2.

[30] O. I. Haugen, 'Safety assurance of complex systems Part 2: Assurance and analysis', DNV AS, Høvik, Norway, Whitepaper, 2019.

[31] G. M. Weinberg, *An Introduction to General Systems Thinking*. Dorset House, 2001, 308 pp., ISBN: 978-0-932633-49-1. Google Books: eU9gDxt9X0wC.

[32] L. Floridi, 'The Method of Levels of Abstraction', *Minds and Machines*, vol. 18, no. 3, pp. 303–329, 1st Sep. 2008, ISSN: 1572-8641. DOI: 10.1007/s11023-008-9113-7.

[33] C. W. Bytheway, *FAST Creativity & Innovation: Rapidly Improving Processes, Product Development and Solving Complex Problems*. Fort Lauderdale, Fla: J. Ross Pub, 2007, 254 pp., ISBN: 978-1-932159-66-0.

[34] O. Haugen, 'Developing a safety argument', in *Demonstrating Safety of Software-Dependent Systems : With Examples from Subsea Electric Technology*, Høvik, Norway: DNV AS, Mar. 2022, pp. 55–82, ISBN: 978-82-515-0324-2.

[35] O. Haugen, 'Safety assurance of complex systems Part 1: Complexity', DNV AS, Høvik, Norway, Whitepaper, 2019.

[36] DNV, *DNV-RP-0684 Dynamic Positioning Systems – systems integration*, Recommended Practice, version March 2025, Mar. 2025.

[37] DNV, *DNV-RP-0671 Assurance of AI-enabled systems*, Recommended Practice, version September 2023, Sep. 2023.

[38] O. I. Haugen, 'Application of ML/MM-SA on a subsea Christmas tree', in *Demonstrating Safety of Software-Dependent Systems; With Examples from Subsea Electric Technology*, T. Myhrvold and M. van der Meulen, Eds., DNV AS, 2022, pp. 321–370, ISBN: 978-82-515-0324-2.

Contribution to the Application of the Adaptive Governance Model to Healthcare Systems

Karim Hardy, PhD
Department of Mathematics, Sciences, and Technology
Embry-Riddle Aeronautical University, Worldwide
Daytona Beach, USA
Karim.hardy@erau.edu

***Abstract*—Healthcare systems increasingly face complex challenges from evolving patient demands, rapid technological advances, and sudden systemic disruptions, such as pandemics or demographic shifts. Traditional governance approaches often rely on static regulations, centralized decision-making, and rigid structures, limiting their effectiveness under dynamic conditions. This article proposes the Adaptive Governance Model (AGM) as an innovative framework to enhance healthcare resilience, safety, and operational efficiency. AGM integrates real-time adaptability, decentralized decision-making, and cross-sector collaboration, supported by artificial intelligence and advanced analytics. The article identifies clear benefits and discusses key implementation challenges by exploring the practical applications of AGM within hospital management, public health crisis response, and emergency medical systems. The practical integration of AI technologies, particularly predictive analytics and neural networks is addressed explicitly. Finally, directions for future research and pilot implementation strategies are proposed to further validate and refine AGM for widespread adoption in healthcare governance.**

Keywords-Adaptive Governance; Healthcare Systems; Resilience; Artificial Intelligence; Emergency Management.

I. INTRODUCTION

Modern healthcare systems are increasingly confronted with multifaceted challenges from demographic shifts, technological advancements, and unforeseen global health crises. The aging population, the prevalence of chronic diseases, and the rapid evolution of medical technologies have collectively intensified the complexity of healthcare delivery and governance [1]. These dynamics necessitate governance models that are both flexible and responsive to the changing landscape.

Often characterized by hierarchical structures and rigid protocols, traditional governance frameworks have demonstrated limitations in effectively managing such complexities. During the COVID-19 pandemic, for instance, many healthcare systems struggled to adapt swiftly to the rapidly evolving situation, highlighting the inadequacies of conventional governance approaches in crisis management [2]. The pandemic underscored the need for governance models that can accommodate uncertainty and facilitate rapid decision-making.

The Adaptive Governance Model (AGM) has emerged as a promising paradigm in response to these challenges. AGM emphasizes flexibility, decentralized decision-making, and stakeholder collaboration, enabling healthcare systems to navigate uncertainties more effectively [3]. AGM offers a framework that aligns with the dynamic nature of contemporary healthcare environments by fostering resilience and adaptability.

This article aims to explore the application of AGM within healthcare systems, addressing the following research questions:

1. How can AGM principles be effectively integrated into healthcare governance structures?
2. What are the potential benefits and challenges associated with implementing AGM in healthcare settings?
3. How does AGM compare to traditional governance models in terms of responsiveness and resilience?

To address these questions, the article is structured as follows: Section II delves into the core principles of AGM and their relevance to healthcare. Section III examines practical applications of AGM in healthcare systems, drawing on case studies and empirical evidence. Section IV discusses the benefits and potential obstacles of AGM implementation. Section V explores the integration of artificial intelligence within the AGM framework. Finally, Section VI concludes with insights and recommendations for future research and practice.

II. CORE PRINCIPLES OF THE ADAPTIVE GOVERNANCE MODEL (AGM)

The Adaptive Governance Model (AGM) offers a dynamic framework tailored to address the complexities inherent in modern healthcare systems. By emphasizing flexibility, decentralization, and collaboration, AGM seeks to enhance healthcare governance's resilience and responsiveness. The model is underpinned by three foundational principles: real-time adaptability, decentralized decision-making, and collaborative resilience.

A. Real-Time Adaptability

Real-time adaptability refers to the capacity of healthcare systems to respond promptly to emerging challenges through continuous monitoring and predictive analytics. This

principle is crucial in environments characterized by rapid changes, such as during pandemics or technological disruptions. Implementing adaptive data governance frameworks enables healthcare organizations to dynamically perceive environmental shifts and recalibrate strategies, accordingly, thereby enhancing systemic resilience [4].

B. Decentralized Decision-Making

Decentralized decision-making involves distributing governance roles across various healthcare system levels to facilitate quicker and more localized responses. This approach empowers local entities to make decisions that are more attuned to specific community needs, thereby improving equity and efficiency. Studies have shown that decentralization can lead to improved retention of healthcare workers and reduced absenteeism, although it requires robust coordination mechanisms to prevent potential drawbacks such as nepotism or resource misallocation [5].

C. Collaborative Resilience

Collaborative resilience emphasizes the integration and coordination among healthcare providers, emergency responders, policymakers, and other stakeholders. By fostering partnerships and shared responsibilities, healthcare systems can better absorb shocks and maintain functionality during crises. Collaborative governance structures have been instrumental in enhancing the adaptive capacity of health systems, particularly in managing public health emergencies [3].

These principles collectively contribute to a more resilient and responsive healthcare governance model, capable of navigating the complexities and uncertainties of contemporary healthcare environments.

III. Applications of AGM in Healthcare Systems

The Adaptive Governance Model (AGM) can significantly enhance healthcare governance by offering strategic flexibility and resilience in various critical areas. Its application can notably improve the effectiveness of hospital network management, public health crisis management, and emergency response systems, each presenting unique demands and complexities.

A. Hospital Network Management

AGM facilitates optimizing resources and improving patient care within hospital networks, especially during periods of high demand. Hospitals can better allocate resources, manage patient flow, and respond to emerging challenges by promoting decentralized decision-making and real-time adaptability. For instance, the integration of adaptive governance strategies has been shown to support networks of local organizational relationships and enable distributed, local control and experimentation, leading to more responsive healthcare services [6].

B. Public Health Crisis Management (Heading 2)

AGM provides a framework for adaptive response strategies in the face of public health emergencies, such as pandemics. By enabling flexible decision-making processes and fostering collaboration among stakeholders, healthcare systems can more effectively manage crises. Research indicates that adaptive governance approaches are crucial for overcoming challenges during health emergencies, as they allow for the alignment of various organizational networks and the scaling of effective interventions [7].

C. Emergency Response Framework

AGM enhances the efficiency of emergency response systems by promoting coordination among medical emergency services, hospitals, and other agencies. Healthcare systems can ensure timely and effective responses to emergencies through collaborative resilience. Studies have highlighted the importance of adaptive governance in disaster preparedness, emphasizing its role in facilitating coordinated efforts and improving overall response capabilities [8].

D. Rationale and Broader Relevance

The application of AGM in these contexts underscores its versatility and effectiveness in enhancing healthcare governance. By embracing the principles of real-time adaptability, decentralized decision-making, and collaborative resilience, healthcare systems can better navigate complexities and uncertainties. The broader relevance of AGM lies in its potential to transform healthcare governance, making it more responsive, efficient, and resilient in the face of evolving challenges.

IV. Benefits and Challenges in Implementing AGM

The Adaptive Governance Model (AGM) offers a transformative approach to healthcare governance, aiming to enhance system resilience, operational efficiency, and crisis responsiveness. However, its implementation also presents challenges that require careful consideration.

A. Benefits

1) Increased Resilience through Adaptive Policies

AGM promotes the development of policies that enable healthcare systems to anticipate, monitor, and respond to disruptions effectively. By fostering adaptive capacity, healthcare organizations can better withstand crises and maintain continuity of care [9].

2) Enhanced Operational Efficiency and Safety

Decentralized decision-making within AGM allows for more localized and timely responses, reducing bottlenecks and improving resource allocation. This approach enhances operational efficiency and patient safety by enabling frontline providers to address issues promptly [6].

3) Improved Crisis Response Capabilities

AGM's emphasis on collaborative resilience facilitates coordinated efforts among healthcare providers, emergency responders, and policymakers. Such integration is crucial for effective crisis management, enabling swift mobilization of resources and sharing information during emergencies [8].

B. Challenges and Recommendations

1) Ensuring Compliance with Healthcare Regulations While Maintaining Adaptability

Balancing the flexibility of AGM with regulatory compliance is a significant challenge. Healthcare organizations must develop governance frameworks that allow for adaptive decision-making while adhering to legal and ethical standards. Implementing robust data governance policies and continuous monitoring can help achieve this balance [4].

2) Addressing Resistance from Traditional Hierarchical Institutions

Transitioning to AGM may face resistance from established hierarchical structures accustomed to centralized control. To mitigate this, organizations should engage stakeholders through transparent communication, provide training on adaptive practices, and demonstrate the benefits of decentralized governance through pilot programs [3].

3) Navigating Ethical Implications of AI-Driven Governance

The integration of AI within AGM raises ethical concerns related to privacy, transparency, and fairness. Ensuring that AI systems are designed with ethical considerations in mind, such as explainability and bias mitigation, is essential. Establishing oversight committees and ethical guidelines can support responsible AI implementation in healthcare governance [10].

V. Practical Integration of Artificial Intelligence in AGM

Integrating Artificial Intelligence (AI) into the Adaptive Governance Model (AGM) offers transformative potential for healthcare systems, enhancing their adaptability, efficiency, and resilience. This section outlines practical methods for incorporating AI into AGM, highlights specific use-cases, recommends appropriate AI tools, and differentiates AGM's AI integration approach from existing healthcare governance models. AGM's integration of AI promotes an ethical framework emphasizing transparency and efficiency in healthcare governance, aligning closely with the principles of adaptive governance and enabling proactive responses to complex health situations [13].

A. Practical Methods for Incorporating AI into AGM

Incorporating AI into AGM involves several strategic steps:

- **Data Infrastructure Enhancement**: Establish robust data collection and management systems to ensure high-quality, real-time data availability.
- **Algorithm Development and Validation**: Develop AI algorithms tailored to specific healthcare needs and ensure they are validated for accuracy and reliability.
- **Integration into Decision-Making Processes**: Embed AI tools into existing decision-making workflows to support real-time adaptability and decentralized governance.
- **Continuous Monitoring and Feedback Loops**: Implement mechanisms for monitoring AI performance on an ongoing basis and incorporate feedback to refine algorithms and processes.

B. Specific Use-Cases for AI in Predictive Analytics and Real-Time Decision-Making

AI's integration into AGM can be exemplified through various use cases:

- **Predictive Disease Modeling**: Utilize AI algorithms to forecast disease outbreaks and patient deterioration, enabling proactive interventions. [19]
- **Resource Allocation Optimization**: Apply AI to predict patient influx and resource needs, facilitating efficient allocation of staff, beds, and equipment. [12]
- **Personalized Treatment Plans**: Leverage AI to analyze patient data and recommend individualized treatment strategies, improving outcomes and patient satisfaction.
- **Emergency Response Coordination**: Integrate AI to streamline communication and coordination among emergency services, hospitals, and other stakeholders during crises. [16]

C. Recommended AI Tools

Selecting appropriate AI tools is crucial for effective integration into AGM:

- **Neural Networks**: Suitable for complex pattern recognition tasks like image analysis and predictive modeling. [11]
- **Decision Trees and Random Forests**: Effective for classification and regression tasks, aiding in diagnostic processes and treatment recommendations.
- **Natural Language Processing (NLP)**: Useful for extracting insights from unstructured data, such as clinical notes and patient feedback.
- **Reinforcement Learning**: Applicable in developing adaptive systems that learn optimal strategies through interaction with the environment.

D. Differentiating AGM's AI Integration Approach

Prioritizing clear actions for AI use in healthcare, AGM differs from traditional healthcare models by specifically outlining strategic priorities to leverage AI effectively, thus maximizing its benefits while addressing ethical and operational concerns [14]. AGM's approach to AI integration distinguishes itself from traditional healthcare governance models in several ways:

- **Emphasis on Adaptability**: AGM prioritizes real-time responsiveness, allowing AI systems to adapt to changing conditions and data inputs dynamically.
- **Decentralized Decision-Making**: Unlike centralized models, AGM supports distributed

governance, enabling localized AI-driven decisions that are context-specific.

- **Collaborative Framework**: AGM fosters collaboration among diverse stakeholders, ensuring that AI integration aligns with the needs and values of all parties involved.
- **Ethical and Transparent AI Use**: AGM incorporates ethical considerations into AI deployment, promoting transparency, accountability, and fairness.

Integrating AI within AGM facilitates ethical, transparent, and efficient governance in healthcare, crucial for proactive decision-making and complex health management [13]. AGM's approach to AI explicitly prioritizes strategic clarity, addressing critical operational and ethical issues that traditional models often overlook [14]. However, while AI presents transformative opportunities for healthcare governance, careful consideration of potential risks, such as bias and data privacy, remains essential [15]. AGM addresses these concerns by emphasizing tools developed specifically with clinician usability and clinical relevance in mind, enhancing practical adoption and operational effectiveness [17]. Additionally, AGM advances a governance approach beyond mere regulatory compliance, adopting adaptive frameworks that thoroughly address privacy, transparency, and fairness, crucial to maintaining stakeholder trust [18]. In terms of predictive analytics, AGM leverages generative AI to dynamically anticipate patient needs and optimize resource allocation, significantly enhancing preparedness and crisis responsiveness in healthcare settings [20].

VI. Conclusion and Future Work

This article presented the Adaptive Governance Model (AGM) as an innovative approach to managing the increasing complexities and challenges contemporary healthcare systems face. Key insights indicate that AGM, characterized by real-time adaptability, decentralized decision-making, and collaborative resilience, holds significant potential for enhancing system responsiveness, resilience, and operational efficiency. Integrating artificial intelligence within AGM further amplifies its effectiveness by enabling predictive analytics, optimized resource allocation, personalized patient care, and coordinated emergency responses.

To move AGM from theory to practice, the next essential steps involve rigorous validation through real-world implementations and targeted pilot projects. Healthcare organizations should initiate carefully designed pilot studies across diverse settings—hospital networks, public health crises, and emergency response systems—to systematically evaluate AGM's practical effectiveness and scalability.

Future research should focus on several critical areas:

1. **Detailed Case Studies**: Conduct in-depth analyses of AGM applications across varied healthcare contexts to document effectiveness, identify potential limitations, and develop best practices.
2. **Practical Demonstrations**: Facilitate real-time demonstrations showcasing AGM's responsiveness and AI-driven decision-making capabilities during simulated or actual healthcare crises.
3. **Expanded Stakeholder Engagement**: Actively engage policymakers, healthcare professionals, technology experts, and patient representatives in iterative dialogues to ensure AGM aligns with real-world needs and ethical standards.

AGM can be refined and broadly implemented by addressing these future research directions, ultimately enhancing healthcare governance systems' resilience and adaptability to emerging global health challenges.

References

[1] World Health Organization, *Global strategy on digital health 2020–2025*, 2021. [Online]. Available: https://www.who.int/publications/i/item/9789240020924

[2] M. E. Kruk, M. Myers, S. T. Varpilah, and B. T. Dahn, "What is a resilient health system? Lessons from Ebola," *The Lancet*, vol. 385, no. 9980, pp. 1910–1912, 2020, doi:10.1016/S0140-6736(15)60755-3.

[3] B. C. Chaffin, H. Gosnell, and B. A. Cosens, "A decade of adaptive governance scholarship: Synthesis and future directions," *Ecology and Society*, vol. 19, no. 3, art. 56, 2014, doi:10.5751/ES-06824-190356.

[4] B. Carroll, "Adaptive data governance in healthcare organizations: A strategic imperative," *LinkedIn Pulse*, 2024. [Online]. Available: https://www.linkedin.com/pulse/adaptive-data-governance-healthcare-organizations-senior-ben-carroll-46tbe

[5] T. Bossert and A. Mitchell, "Health sector decentralization and local decision-making: Decision space, institutional capacities and accountability in Pakistan," *Social Science & Medicine*, vol. 72, no. 1, pp. 39–48, 2011, doi:10.1016/j.socscimed.2010.10.019.

[6] A. Best, J. E. Saul, S. Carroll, and J. Bitz, "Building integrated, adaptive and responsive healthcare systems," *BMC Health Services Research*, vol. 22, no. 1, pp. 1–12, 2022, doi:10.1186/s12913-022-07856-z.

[7] M. Khan, P. Roy, and R. Chowdhury, "An adaptive governance and health system response for the COVID-19 emergency," *World Development*, vol. 137, art. 105213, 2020, doi:10.1016/j.worlddev.2020.105213.

[8] K. Evans, "Adaptive crisis management: Holistic frameworks in emergency medical ecosystems," 2024. [Online]. Available: https://drkerryevans.com/emergency-medicine/adaptive-crisis-management-holistic-frameworks-in-emergency-medical-ecosystems/

[9] A. Smaggus, J. C. Long, L. A. Ellis, R. Clay-Williams, J. Braithwaite, and R. L. Wears, "Government actions and their relation to resilience in healthcare during the COVID-19 pandemic in New South Wales, Australia and Ontario, Canada," *International Journal for Quality in Health Care*, vol. 34, no. 1, art. mzab122, 2022, doi:10.1093/intqhc/mzab122.

[10] HITRUST Alliance, "The ethics of AI in healthcare," 2023. [Online]. Available: https://hitrustalliance.net/blog/the-ethics-of-ai-in-healthcare

[11] E. Choi, M. T. Bahadori, A. Schuetz, W. F. Stewart, and J. Sun, "Doctor AI: Predicting clinical events via recurrent

neural networks," *arXiv preprint arXiv:1511.05942*, 2016. [Online]. Available: https://arxiv.org/abs/1511.05942

[12] Intuz, "AI predictive analytics in healthcare: Key benefits and use cases," 2024. [Online]. Available: https://www.intuz.com/blog/use-cases-ai-predictive-analytics-in-healthcare

[13] Frontiers in Digital Health, "AI with agency: A vision for adaptive, efficient, and ethical healthcare," 2025. [Online]. Available: https://www.frontiersin.org/journals/digital-health/articles/10.3389/fdgth.2025.1600216/abstract

[14] Health Affairs, "Artificial intelligence in health and health care: Priorities for action," 2024. [Online]. Available: https://www.healthaffairs.org/doi/10.1377/hlthaff.2024.01003

[15] ScienceDirect, "Artificial intelligence in healthcare delivery: Prospects and pitfalls," 2025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2949916X24000616

[16] PMC, "Artificial intelligence in healthcare: Transforming the practice of medicine," 2021. [Online]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC8285156/

[17] BMC Medicine, "Artificial intelligence tool development: What clinicians need to know," 2025. [Online]. Available: https://bmcmedicine.biomedcentral.com/articles/10.1186/s12916-025-04076-0

[18] International Association of Privacy Professionals, "Beyond compliance: The case for adaptive AI governance," 2025. [Online]. Available: https://iapp.org/news/a/beyond-compliance-the-case-for-adaptive-ai-governance

[19] Innovation News Network, "The role of AI in predicting and managing disease outbreaks," 2024. [Online]. Available: https://www.innovationnewsnetwork.com/the-role-of-ai-in-predicting-and-managing-disease-outbreaks/49277/

[20] Confluent, "Predictive analytics in healthcare: Using generative AI and Confluent," 2025. [Online]. Available: https://www.confluent.io/blog/predictive-analytics-healthcare/

 ISBN: 978-1-68558-280-7

An End-to-End Method for Operationalizing Trustwothiness in AI-Based Critical Systems

Karla Quintero, Lucas Mattioli, Henri Sohier
IRT SystemX, France
email: {karla.quintero, lucas.mattioli, henri.sohier}@irt-systemx.fr

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

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

Keywords-Trustworthy AI, safety-critical AI-based systems, end-to-end engineering of AI-based processes, trustworthiness attributes.

I. Introduction

Any technology, even Artificial Intelligence (AI), is developed to provide a service fulfilling some needs. In our context, an AI-based system is defined as a system that incorporates software-based AI components. AI-based critical systems, which can have severe consequences in case of failure, are considered to be "high risk" under the EU AI Act [1]. These systems can for example represent safety components of regulated products which are required to undergo a third-party conformity assessment. Examples of such systems can be found in the fields of transportation, healthcare, defense, and security in general. The deployment of such systems is contingent upon their demonstrated capacity to deliver the anticipated service in a secure manner, while meeting user expectations with regard to quality and continuity of service. Furthermore, users might consider as negative any surprising or unexpected actions from the system.

In order to characterize such systems with a view to quality assurance, [2] proposed considering several dimensions: the artifact type dimension, the process dimension, and the trustworthiness characteristics attributes that are relevant to software product or system quality. In addition, software quality is at the center of the SQuaRE (Systems and Software Quality Requirements and Evaluation) series of standards, and the specific nature of AI is addressed more specifically in order to offer a quality model for AI systems. Consequently, the design of AI-based critical systems necessitates the demonstration of their trustworthiness, as asserted by [3].

Trustworthy AI is based on three components [4], which should be met throughout the system's entire life cycle: firstly, it should be lawful, in that it complies with all applicable laws and regulations; secondly, it should be ethical, ensuring adherence to ethical principles and values; and thirdly, it should be robust, both from a technical and social perspective since, even with good intentions, AI systems can cause unintentional harm. Thus, to support the industrial design of such systems, there is a requirement for Trustworthy AI Engineering, a new discipline that is an evolving multi-disciplinary field. The aim of this discipline is to ensure that an AI-based critical system (in the safety, mission and business domains) is valid, explainable, resilient, safe, secure, compliant with respect to regulation, standardization, and responsible practices (ethical and sustainable). When dealing with critical systems, several additional constraints must be considered. In the context of system design, there is a need to optimize processes, provide justification, replicate where possible, and implement improvements. However, it is also essential to ensure that the system meets the appropriate level of trustworthiness [5]. This includes robustness (defined as the ability of a system to withstand errors during execution and to cope with erroneous input), cyber-security, and dependability (including reliability, availability, maintainability, and safety properties), among others.

Thus, in the following, we will first remind the today context of AI regulation and standardization as *"trustworthiness is the ability to meet stakeholders' expectations in a verifiable way"*. Then, we present an end-to-end methodology to support "Trustworthy AI Engineering", which encompasses the entire lifecycle of AI-based systems, from Operational Design Domain (ODD) specification to maintenance. This methodology covers data engineering, algorithm design, development, deployment and monitoring. This systematic approach involves organizing multi-disciplinary and fragmented approaches to trusted AI and applying a continuous workflow approach. Measures to improve AI trustworthiness must be taken at every stage, such as data sanitisation, robust algorithms, anomaly monitoring and risk auditing.

II. Regulation and Standardization

Ensuring safety, reliability, availability and maintainability, means AI systems must perform and continue to perform as intended under sufficient conditions. Hazard analysis and risk assessment are tailored to the unique characteristics of AI. These include potential critical errors in training data or knowledge representation, and the ability of the AI model to generalize to unseen, operational data. The performance requirements on the AI algorithm are often driven by safety

 ISBN: 978-1-68558-280-7

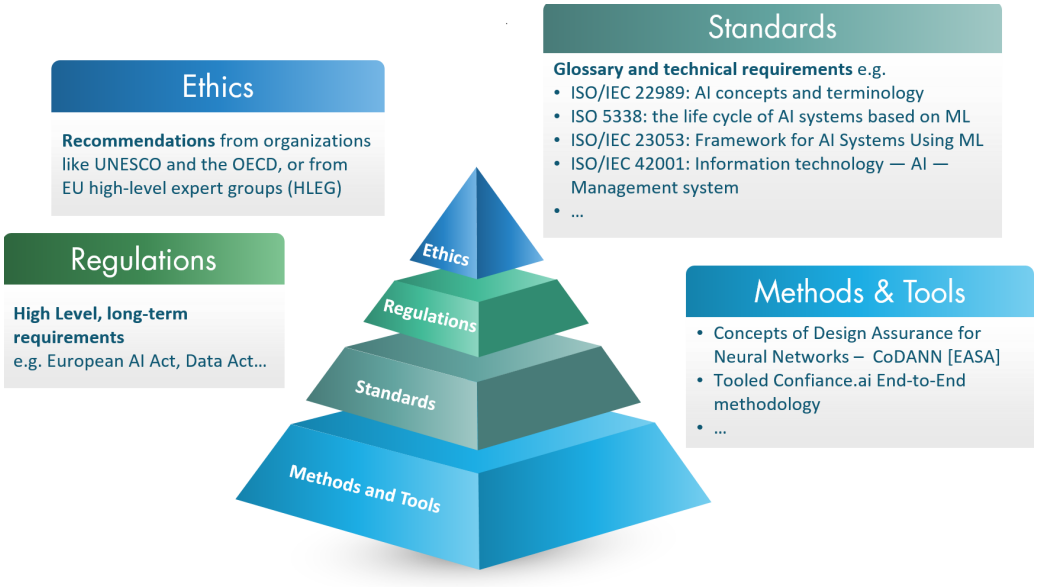

Figure 1. From ethics to the end-to-end methodology through regulation and standards

objectives to limit its worst credible approximation error to a given acceptable threshold.

However, trustworthiness is tightly related to accountability: accountability can be considered as a factor of trust or as an alternative to trust. Then, in [6], dependability is used to represent the overall quality measure of a system based on four sub-attributes including security, safety, reliability, and maintainability. Thereafter, security and dependability became key attributes for computer-based system trust [7].

In 2019, the U.S. National Artificial Intelligence Research and Development Strategic Plan [8] emphasized that: *"standard metrics are needed to define quantifiable measures in order to characterize AI technologies"*. More recently, [9] noted that "*significant work is needed to establish what appropriate metrics should be to assess system performance across attributes for responsible AI and across profiles for particular applications/contexts.*".

Governments are responding with regulations typically associated to human rights. In 2024, the European Union adopted the AI Act. These regulations set high-level, long-term requirements, sometimes building on recommendations from organizations like UNESCO [10] and the OECD [11], [12], or from High-Level Expert Groups (HLEG) [4].

These high-level requirements require to be operationalized for companies and developers. As shown in figure 1, standards and regulation frameworks define more detailed requirements but remain focused on *what* to do rather than *how* to do it, leaving the choice of a tooled end to end methodology to use for the development of AIs fulfilling these requirements.

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

In the aeronautic domain, EASA [13] proposes a model of trustworthiness based on: the characterization of the Machine Learning (ML) application (high-level function/task, concept of operations, functional analysis, classification of the ML application), safety assessment, information security management, and ethics-based assessment (which includes the 7 pillars of the ALTAI [14]).

The Fraunhofer [15] offered an analysis of the standard [16, Under development] on management system for AI, stating compliance to the standard can contribute to ensuring AI trustworthiness since it encompasses the pillars of the ALTAI, provided that a third-party verification has been performed and along with an adapted quality management system.

In the same period, the NIST produced an analysis of the components of trust [17] and highlighted several top level aspects for the design of a trustworthiness model, that should encompass the user experience, the perceived technical trustworthiness, the pertinence of each trustworthiness characteristic in the user's specific context of use...

Moreover, ETSI set-up in 2019 an Industry Specification Group on Securing AI (ISG SAI) from attack to resilience [18] providing existing and potential mitigation against threats for AI-based systems.

Robust security measures must protect AI systems from cyberattacks, data breaches, and unauthorized manipulation. These measures should include advanced threat detection and mitigation strategies and resilience mechanisms to operate securely in hostile environments. Cybersecurity should be embedded in the system and data pipelines. The lines between security and safety are not always clear when it comes to AI.

 ISBN: 978-1-68558-280-7

Incorrect outputs can be caused by malicious actions or natural events.

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

However, it is imperative to recognize that the transfer of AI technology, particularly Machine Learning (ML), must align with specific standards and processes to ensure the successful transformation of research outcomes into industrial products that are fit for the intended purpose and meet customer needs. For instance, as data collection and analysis are pivotal for the development of any ML-based system, it is essential to prioritize the data quality. This necessitates adherence to compliance regulations (such as data privacy). Concurrently, operational requirements encompassing the maintenance must be addressed. Consequently, it is evident that the development and implementation of AI/ML systems is a multifaceted process involving both technical and business aspects, from problem conception to delivery to customers. Consequently, the development and operation of AI-based critical systems necessitates the utilization of an end-to-end tool-based AI engineering methodology, which will be subsequently delineated.

III. The end-to-end methodology

The version of the methodology presented herein has been produced as a result of the work within the Confiance.ai program [19] [20], [21] and the associated roadmap is nourished by industrial needs and the evolution of the state-of-the-art [22]. Namely, several industrial projects and research initiatives have derived from Confiance.ai, generating the emergence of an ecosystem for the engineering of trustworthy AI for critical systems. The proposed end-to-end methodology addresses the following challenges [23]:

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

By addressing these challenges, the end-to-end methodology aims to answer the research question: How to ensure the reliability and trustworthiness of AI-based safety-critical systems? It is based on the premise that the development of ML-based critical systems should be structured with a trustworthiness imperative from the design phase, thereby providing precise requirements for integration, verification, and validation, as well as for proper deployment and maintenance [24]. It is a multi-domain collaboration that leverages concepts and procedures coming from different fields into the agnostic proposal of engineering trustworthy ML-based critical systems. The result is the formalization, through a common language, of the structure and workflow for all actors involved in the process of designing trustworthy ML-based critical systems, i.e. data engineers, systems engineers, safety engineers, software engineers, among many others.

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

Developing ML-based systems can be visualized as a "W-shaped" life-cycle (see figure 2). This W-shape can be split into two parts. For AI systems, "intended goal"/"intended purpose" and "intended domain of use" are very high-level requirements that have to be translated into "engineering terms". The engineered "intended domain of use" is called Operational Design Domain (ODD). The ODD is the operational conditions for which an AI system is specified, designed, verified, assessed, operated, and disposed. ML engineering life-cycle begins with defining AI/ML algorithm requirements refined from system specification. This ML specification step includes the characterization of the ODD.

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

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

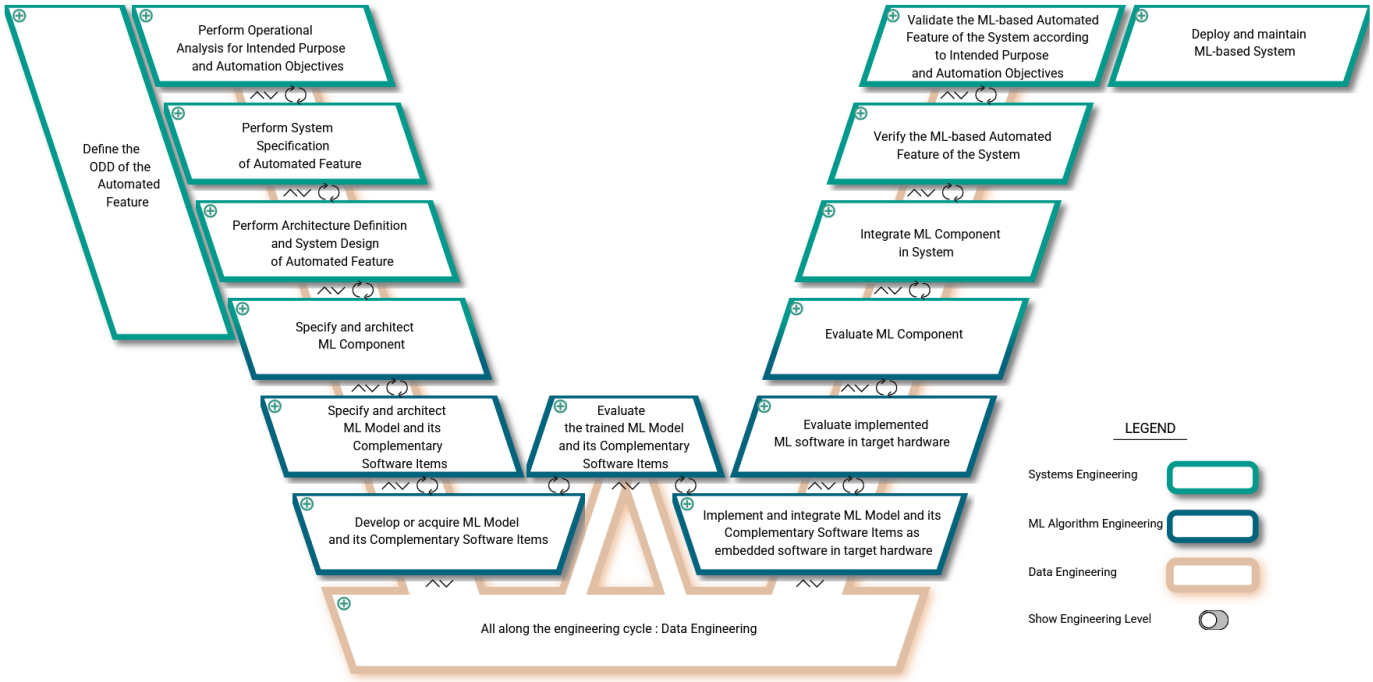

Figure 2. High-level view of the end-to-end methodology.

computational efficiency and performance.

The second "V" of the "W-shaped" life-cycle includes the implementation engineering processes performed on the target platform (e.g., specific hardware embedded in a ground or aerial vehicle). Validation and verification activities are driven by key trustworthiness properties, specified in low-level ML requirements. Validation activities ensure the correctness and completeness of ML requirements by verifying, analyzing and tracing them back to higher-level requirements. Verification activities include simulating extensively, testing edge/corner robustness, scenario-based testing, analyzing the ML model explainability and ODD coverage analysis. The first level of verification ends with a selected AI model, which meets all its requirements in the development (learning) environment and serves as a design specification, ready for implementation into software and/or complex electronic hardware elements in the second level of verification. figure 3 shows a high-level view of the verification phase of an ML-based automated feature and the interaction with specification and validation phases.

MLOps, or Machine Learning Operations, and AM/ML Engineering, while closely related, serve distinct roles within the machine learning lifecycle. MLOps focuses on the operationalization of machine learning models, ensuring that they are deployed efficiently and maintained effectively in production environments. In contrast, ML Engineering is primarily concerned with the development and the maintenance of an AI-based system. Thus MLOps emphasizes the operational aspects of machine learning, while ML/AI Engineering is centered on the overall lifecycle of the system covering all system engineering concerns (from specification to maintenance) which includes MLOps. MLOps involves collaboration between data scientists, ML engineers, and IT operations teams when AI/ML Engineering involves system ad software engineers, data scientists, safety and cyber-security engineers. The end-to-end methodology (see figure 2) supports all AI/ML engineering activities where MLOps covers ML algorithm engineering and data engineering.

IV. Trustworthiness Attributes and Assessment

Trustworthiness is fundamental for the successful development and adoption of AI-based critical systems. Thus, trustworthiness assessment [25] can be defined as the process of evaluating and determining the level of trustworthiness of a given characteristic, such as robustness [26], accuracy, reliability [20], or effectiveness, in the context of AI systems engineering.

Nevertheless, it is very misleading to only judge how good an AI system is based on how accurate it is. It is also difficult to test and check the quality of software in the traditional way, and it is even difficult to measure test coverage at all. Trust and trustworthiness are complex, and so one of the main issues we face is to establish objective attributes such as accountability, accuracy, controllability, correctness, data quality, reliability, resilience, robustness, safety, security, transparency, explainability, fairness, privacy, and compliance with regulatory actors. We need to map these attributes onto the AI processes and its lifecycle and provide methods and tools to assess them. This highlights the importance of quality requirements, which are non-functional requirements and are particularly challenging in AI systems, although many of them can be considered in any critical system. Furthermore, this can also include risk and process considerations. The attributes

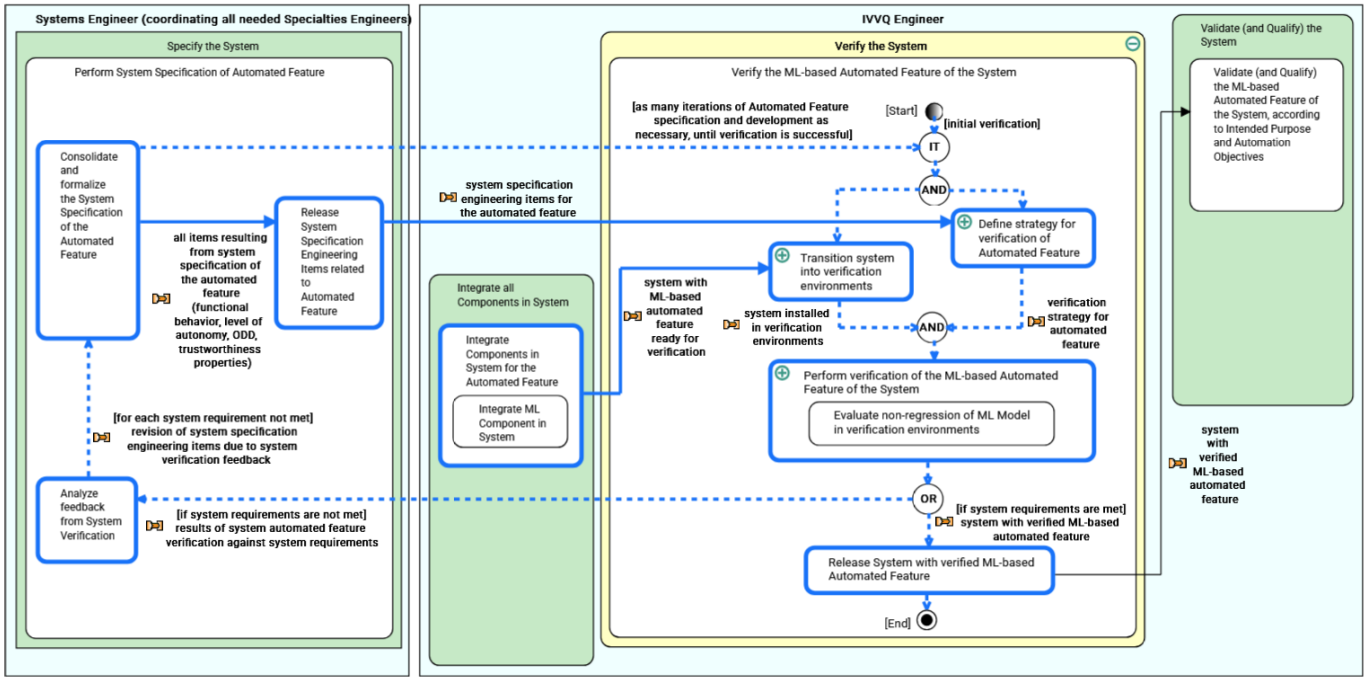

Figure 3. Verification Phase: verification of the ML-based automated feature of the system.

and values for these requirements depend on things like how important the application is, what the AI system is used for, how it will be used, and the people involved. So, in some situations, some attributes may be more important than others, and new attributes may be added to the list [27]. Clear specifications of the non-functional requirements will help clarify these conflicts and can also encourage innovation that solves some of these conflicts, allowing us to fulfill more of them at the same time.

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

- **Robustness**. Robust AI systems should be resilient to various perturbations (ie: variations in input data and operating conditions). This requires :
  - Adversarial robustness, ensuring the system is not easily manipulable by adversarial attacks.
  - OOD Robustness (Out-Of Distribution), the system must generalize well across different environment and be trained on diverse datasets.
  - Model monitoring, ensuring a continuous evaluation of the AI models, to detect performance degradation.

  Two types of strategies for robustness by design can be distinguished: empirical robustness and formal robustness.
  - Empirical methods emphasize on uncertainty quantification and adversarial robustness of ML Models, like the adversarial training method.
  - Formal methods aim to design neural networks with exact robustness guarantee such that, under some constraints on the norm of the perturbation added to the input, the class of the input remains the same for the ML Model. Lipschitz method is one example of formal methods advocated as enablers for robustness by design.
- **Explainability, Interpretability and Comprehensibility**. Trustworthy AI should be transparent and its decisions should be interpretable where
  - Explainability deals with the capability to provide the human with relevant information on how an AI application is coming to its result.
  - Interpretability relates to the capability of an element representation (an object, a relation, a property...) to be associated with the mental model of a human being. It is a basic requirement for an explanation.
  - Comprehensibility refers to the capability of an element representation (an object, a relation, a property...) to be understood by a person according to its level of expertise or background knowledge.

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

  There is a profusion of methods, tools, and solutions available, each with its own set of advantages, drawbacks, and trade-offs [28]. The many different approaches show how tricky it is to make sure that AI and machine learning models can explain their predictions and decisions. Choosing the

right way to make models explainable is a technical and strategic decision. It depends on the unique needs and limits of the people it will be used by, the specific example it will be used for, and the wider situation in which the AI system will be used. What works for a medical diagnosis model may not work for the aeronautic domain, and what regulators expect can be very different from what end-users or business stakeholders expect. The Confiance.ai program provides a "Methodological Guideline for Explainability" (https://catalog.confiance.ai/) which is designed to be a complete guide to help people use AI. It will explain why explainability is important, highlight the many available methods, and offer guidance on selecting the most suitable approach based on the specific situation.

- **Fairness and Bias Mitigation**. AI models should be free from discriminatory biases. This involves:
  - Bias detection and correction techniques, in the data processing and model training phases.
  - Regulatory alignment with fairness standards (eg: GDPR, AI Act).
- **Safety and Security**. An AI-based system must meet rigorous safety and security requirements:
  - Safety analysis and certification based on standards.
  - Cybersecurity counter-measures, integrated on the AI pipeline.

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

- **Operational Design Domain (ODD) definition**
  - Define the operational boundaries where the AI system is expected to function reliably.
  - Establish clear environmental constraints for the AI-system's development.

  The ODD is a description of measurable foreseeable operating conditions within which a system/component shall operate. A traceability property shall be assured between the different levels of ODD (system, subsystem or component).
- **Systems Engineering**
  - Ensure AI system-level requirements are defined in alignment with overall system objectives.
  - Align AI-based system requirements with preexisting system engineering standards and certification guidelines.
- **Data Engineering**
  - Rely on a robust data pipeline to guarantee data integrity, consistency, and traceability across the engineering cycle.
  - Implement bias mitigation strategies at the data collection and processing stages.
  - Use adaptive data augmentation strategies to improve data diversity and model generalization to distribution shifts and operational scenarios.
- **ML Algorithm Engineering**
  - Use ML robustness techniques, designed to handle perturbation and adversarial outputs.
  - Incorporate explanability techniques to have understandable decisions.
  - Apply Uncertainty quantification techniques to asses the model's confidence.
- **Verification and Validation**
  - Perform extensive simulation-based testing to asses performances under edge cases.

In addition, measuring how trustworthy AI systems are is tricky. The ideas behind them are complicated, the characteristics they produce are different, and you can't always compare them. The Confiance.ai program proposes an innovative way to measure trustworthiness using (max,+) algebra [29] based on a complete hierarchical model that brings together different properties, such as how strong, effective, dependable, easy to use and human agency, and human oversight) into a single assessment method. This offers advantages over traditional weighted averaging methods by better handling extreme values and preserving sensitivity to critical indicators, while maintaining sensitivity to critical indicators to provide detailed, understandable assessments of AI-based system trustworthiness.

V. Conclusion and future works

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

These experiments have demonstrated the importance of integrating diverse tools and methods to address expectations regarding trusted ownership, as illustrated by the following two examples: In a use case involving autonomous driving, the analysis of dataset diversity reveals a limited presence of night-time images, prompting the generation of synthetic night-time data. This data exhibits a 'domain gap' and undergoes "domain adaptation" prior to integration into the model training data. These tools, instrumental in the construction of data sets, will also be reused in the supervision stage of the use case. In an aeronautical use case called LARD for "Landing Approach Runway Detection" [30] and represented figure 4, a data quality supervision module is incorporated to consolidate the confidence score of an ML model (see figure 4). In this example, local image quality estimators (e.g. level of blur, brightness) are taken into account in the detection zone of the landing strip that is being detected. The combination of these indicators with the other indicators intrinsic to the model facilitates the establishment of a level of confidence for

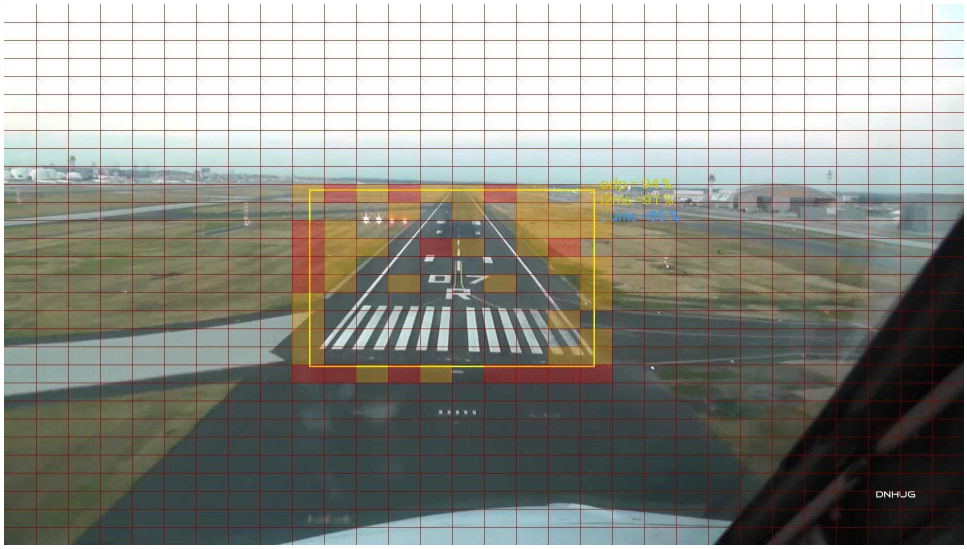

Figure 4. Example of the implementation of a supervision tool on the LARD use-case

the system component. In addition to providing a numerical value, this implementation serves as a tool to facilitate the interpretation of model and data errors.

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

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

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

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

ACKNOWLEDGMENT

This work has been supported by the French government under the "France 2030" program, as part of the SystemX Technological Research Institute within the Confiance.ai Program (www.confiance.ai).

REFERENCES

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

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

[3] H. Liu *et al.*, "Trustworthy AI: A computational perspective", *ACM Transactions on Intelligent Systems and Technology*, vol. 14, no. 1, pp. 1–59, 2022.

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

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

[6] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing", *IEEE transactions on dependable and secure computing*, vol. 1, no. 1, pp. 11–33, 2004.

[7] J.-H. Cho *et al.*, "Stram: Measuring the trustworthiness of computer-based systems", *ACM Computing Surveys (CSUR)*, vol. 51, no. 6, pp. 1–47, 2019.

[8] NSTC, *The national artificial intelligence research and development strategic plan: 2019 update*. National Science and Technology Council (US), 2019.

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

[10] UNESCO, "Recommendation on the Ethics of Artificial Intelligence", Tech. Rep. SHS/BIO/PI/2021/1, 2022.

[11] OECD, "Recommendation of the Council on Artificial Intelligence", Legal Instruments, May 2019.

[12] OCDE, *G7 Hiroshima Process on Generative Artificial Intelligence (AI)*. 2023, p. 37. DOI: https://doi.org/https://doi.org/10.1787/bf3c0c60-en.

[13] EASA, *Concept Paper First Usable Guidance for Level 1 Machine Learning Applications*, 2021.

[14] P. Ala-Pietilä *et al.*, *The assessment list for trustworthy artificial intelligence (ALTAI)*. European Commission, 2020.

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

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

[17] B. Stanton, T. Jensen, *et al.*, "Trust and artificial intelligence", *preprint*, vol. 10, 2021.

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

[19] B. Braunschweig, R. Gelin, and F. Terrier, "The wall of safety for ai: Approaches in the confiance. ai program", in *Workshop on Artificial Intelligence Safety (SAFEAI)*, 2022.

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

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

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

[23] A. Awadid, X. Le Roux, B. Robert, M. Adedjouma, and E. Jenn, "Ensuring the reliability of ai systems through methodological processes", in *2024 IEEE 24th International Conference on Software Quality, Reliability and Security (QRS)*, IEEE, 2024, pp. 139–146.

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

[25] B. Braunschweig *et al.*, "AITA: AI trustworthiness assessment: AAAI spring symposium 2023", *AI and Ethics*, vol. 4, no. 1, pp. 1–3, 2024.

[26] K. Kapusta, L. Mattioli, B. Addad, and M. Lansari, "Protecting ownership rights of ml models using watermarking in the light of adversarial attacks", *AI and Ethics*, vol. 4, no. 1, pp. 95–103, 2024.

[27] J. Mattioli *et al.*, "An overview of key trustworthiness attributes and kpis for trusted ml-based systems engineering", *AI and Ethics*, vol. 4, no. 1, pp. 15–25, 2024.

[28] S. Naveed, G. Stevens, and D. Robin-Kern, "An overview of the empirical evaluation of explainable ai (xai): A comprehensive guideline for user-centered evaluation in xai", *Applied Sciences*, vol. 14, no. 23, p. 11 288, 2024.

[29] J. Mattioli, M. Gonzalez, L. Mattioli, K. Quintero, and H. Sohier, "Leveraging tropical algebra to assess trustworthy ai", in *Proceedings of the AAAI Symposium Series*, vol. 4, 2024, pp. 81–88.

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

The Hidden Business Costs of Ignoring Performance Testing - The Silent Budget Killer

Sowmya Chintakindi
Independent Researcher and Sr. Performance Engineer, USA
e-mail: sowmyar909@gmail.com

***Abstract*—In this era of artificial intelligence and the digital world, the speed and responsiveness of the application have become a critical factor in maintaining a competitive edge. With increasing user expectations of a seamless and high-performance application, even minor delays can lead to customer dissatisfaction and may drive them to competitors. This increase in user expectations made performance testing one of the crucial aspects of the software development life cycle to evaluate the application's speed, reliability, and responsiveness under varying load conditions. Despite its critical role, organizations often overlook performance testing until failure strikes, users leave, and revenue is lost. This paper aims to raise awareness of the importance of performance testing and the consequences of ignoring it. Through real-world case studies and industry insights from practical experience, this paper highlights the impact of inadequate performance testing on the business. Also, it explores best practices to make applications scalable, reliable, and efficient. In a world where every milliseconds matters, performance testing shouldn't be an option - it's a necessity.**

Keywords-Performance testing; Reliability; User experience; Hidden costs.

I. INTRODUCTION

Performance testing is typically positioned at the final stage after development and functional testing. Due to this, it frequently receives limited time and attention as teams spend most of the time in development and validation. The common assumption is that bypassing performance testing can save time and accelerate deployment if no significant performance-related changes are made. However, this will be done at a hidden cost. The actual cost of this may not be immediate. Still, the risks accumulate beneath the surface in unexpected outages, slowness, business loss, and frustrated customers who may never return. This paper explores the hidden costs of bypassing performance testing that organizations cannot afford to ignore and provides some strategies to have seamless and resilient applications.

This paper starts with research methodology in section 2 and then provides background on performance testing, how it is performed, and how it helps businesses in section 3. Section 4 provides some understanding of IT outages, their causes, effects, and preventive measures. This is followed by some case studies on applications that were affected by bypassing performance testing and the loss incurred in section 5. Conclusions are drawn in section 6.

II. RESEARCH METHODOLOGY

This study captured some of the real-time case studies to analyze the importance of performance testing. This research highlights that even minor modifications can impact overall performance and potentially lead to revenue loss. This study demonstrated that continuous and thorough performance validation is essential for maintaining system reliability and business outcomes.

A. Description and purpose of the paper

This study highlights the importance of implementing performance tests and active monitoring practices from early development to production deployment. It is designed to raise awareness, guide organizations, and advocate for a performance-first mindset to avoid unanticipated business losses [1].

B. Research questions

1) RQ1: What happens when performance testing is ignored before a release?

Rationale: When performance testing is ignored before a release, applications will be at risk in production with slow response times, system crashes under load, and loss of business. This study aims to bring awareness to how important it is to have performance tests for the changes made.

2) RQ2: Why do some teams bypass performance testing?

Rationale: Software teams often overlook performance testing. Assuming that there were no changes related to performance, the quality assurance team functionally tested the software, and no performance-related issues would arise. This study highlights the most common reason for performance bottlenecks: bypassing performance testing.

3) RQ3: How do performance-related failures affect customer trust and brand reputation? Rationale: When performance issues surface, it can cause poor customer experience and unpredicted revenue loss. Ultimately, this can damage the company's reputation and reduce customer trust. This study identified some industries that were affected due to performance issues.

4) RQ4: What strategies can be implemented to evaluate business costs of performance issues? Rationale: Since this paper highlighted some of the performance issues that affected some industries, it also mentions the strategies to implement performance testing for resilient systems.

5) RQ5: What are the gaps between state-of-the-art and state-of-practice in performance testing? Rationale: This study helps identify the importance of performance testing and the impact of ignoring it, which is not found in other scholarly articles [2][3][4].

C. Limitation of the approach

Though performance testing is critical, it has some limitations. Setting up realistic test environments can be complex.

Performance testing can sometimes produce negative results if the application's configuration or capacity is not equivalent to production. Proper planning and analysis may lead to misinformation and misguided optimizations .

III. Performance Testing

As internet users are increasing, so does the load on applications. We need performance testing to maintain the applications to perform efficiently and effectively with minimum infrastructure.

A. What is performance testing?

UptimeIntelligenceIt is one of the critical processes in the Software development life cycle that ensures the system is stable, reliable, and scalable under various load conditions. This testing simulates user load from routine traffic to surviving extreme stress. This uncovers how the system truly performs, whether it's measuring response times or testing under peak load conditions; performance testing uncovers hidden bottlenecks, fine-tuning the application, and guarantees system stability to provide exceptional user experience at every turn.

B. Evolution of performance testing

In the early 1990s, as the Internet began to gain attention, performance testing was purely a manual endeavor. Testers relied on manual approaches to measure application performance. In 1991, Mercury introduced WinRunner, an automated GUI testing tool that allows users to record and replay user activities. This reduced significant reliance on manual testing.

The demand for faster applications grew as the Internet boomed, making performance testing more essential. In this momentum, Mercury developed the first performance testing tool, LoadRunner, in 1993. This tool helped testers assess application performance under heavy loads. Since then, performance testing has continuously evolved, with many tools emerging into the market. The rise of open-source load testing tools enabled organizations to execute performance testing more efficiently and cost-effectively.

Performance testing has evolved to integrate seamlessly with Agile methodology and DevOps to emulate continuous integration and deployment models. The advent of cloud platforms has further enabled performance testing to evolve to provide scalable environments. Today, integration with AI has enhanced this process to be quicker and more proactive.

C. Performance testing process

Performance testing is a structured process, and its life cycle includes various phases. Starting with nonfunctional requirements gathering, test planning, test case creation, test script creation, execution, result analysis, and dashboard generation.

1) Non-functional requirements gathering: This process begins with gathering non-functional requirements such as expected throughput, critical business transactions, response times, and anticipated resource utilization. Multiple meetings are necessary to collect these requirements. Understanding these from a business and technical perspective helps plan effective testing activities.

2) Test Plan Creation: The next step in this process is creating a test plan and test cases. A test plan is a comprehensive document summarizing all the requirements gathered in the first step of the process. Creating these test plans ensures the effective execution of performance testing. Test case documents list all the scenarios that need to be tested.

3) Test Script Creation: Tests are created using testing tools like LoadRunner, JMeter where different test scenarios are created to simulate user actions virtually and real-time load conditions.

4) Test Execution: Next is a test environment to execute performance tests in the lab. This is the crucial step, as this lab needs to be a production replica to ensure accurate test results. The execution phase starts once the test scripts are ready. This step requires active monitoring of applications with monitoring tools during the test to find bottlenecks or areas of performance improvement. This is where the test plan will effectively plan the number of tests and duration of the execution phase.

5) Results Analysis and Report generation: Once the tests are completed, all the test results are gathered, and a summary report is generated. In some cases, tests are executed again after performance improvements.

D. Key performance metrics

Key performance metrics are used to evaluate the application's performance and efficiency during performance testing. The following are some of the key performance indicators that are captured during performance testing to assess application performance and find bottlenecks to enhance the system.

1) Response times: It is the measure of time taken for a system to respond to a user request. It is calculated by averaging the response times of all the requests sent during the test. In some cases, the 90th percentile of the response times was measured. The 90th percentile response time is calculated as the average response time corresponding to the fastest 90% of the requests.

2) User load: It refers to the number of virtual users simulated with a load-testing tool to access the system under various real-time conditions. These conditions can include average load, peak load, stress load, and concurrent user load.

3) System utilization of the server: This refers to the percentage of system resources like CPU, memory, disk, and network used during the performance test. It provides information about how well the system handles the load during the test.

4) Latency: This refers to the delay between sending a request to a server and receiving a response from the server. This latency can be in the system, network, disk, or application. This metric is measured as time to first buffer, network latency, or round-trip time.

5) Error rate: It is the measure of the percentage of failed requests out of the total requests sent during the test. It is measured as transaction error rate, HTTP error codes, and Network error rate. The higher the error rate, the more unreliable the system is.

6) Page loading time: This is the metric measured for web-based applications. It measures the time taken to load the page, all the image files, DNS lookup, connection time, and server processing time during the test.

7) Page size: This is the metric used for web-based applications. It measures the size of the page, including image files, HTML, and non-HTML resources.

8) database metrics: These are the metrics obtained from the database, such as long-running queries, top SQL, deadlocks, IO, and many more.

E. Behind the scenes of performance testing

Relying on the same number of physical computers to generate hundreds or thousands of users seems impractical. Instead, load testing tools can enable this user load simulation virtually with just a few machines. Figure 1 is the architecture of a typical load testing environment where virtual users are simulated and executed performance tests like real-world scenarios.

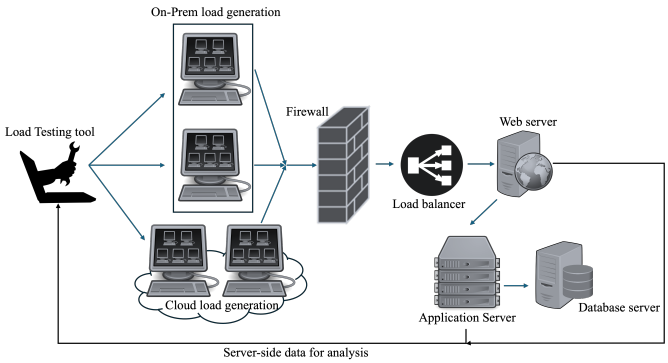

Figure 1. Architecture of a typical load testing environment

The load testing tool uses load generator resources to simulate virtual users to send traffic over the firewall to the web server or application servers like real users and receive responses and metrics for further analysis.

F. Business outcome of performance testing

Performance testing not only improves performance of the application but directly impacts business success.

- Faster response and smooth display, keep users to stay.
- Seamless application increases customer satisfaction and attracts new users.
- As user engagement increases, business growth accelerates and thus higher revenue.
- Organizations with high performance applications gain a competitive edge.
- It helps organization to identify bottlenecks early and prepare for any unplanned application failures.
- Performance testing prevents outages and failures that damage customer trust.
- It contributes to sustainability by optimizing resource utilization and energy consumption to create eco-friendly and cost effective digital solutions.

IV. Understanding IT outages: Causes, Effects and Preventive measures

As technology continues to evolve, the reliability on software is rapidly growing and organizations continue to face significant challenges in maintaining up time of the systems.

A. Causes of IT outages

A recent data from Uptime Intelligence, on average, there are 10 to 20 high-profile IT outages every year that cause serious or severe financial loss [5].

Another study from Magnita reveals that 68% of the organizations conduct performance testing, and 55% of them encounter difficulties due to the unavailability of test environments. This indicates that over half of the organizations may deploy software into production without proper testing and mainly performance testing to assess the system reliability under real-world conditions [6].

IT outages can occur from a variety of factors like hardware failures, cyberattacks, software faults, and capacity or congestion-related issues, which contribute to 22% of IT outages, according to respondents from Statista. This reveals that nearly a quarter of the IT outages occurred due to poor handling of demand, resulting in performance degradation [7].

B. Effects of IT outages

According to survey conducted by uptime, In 2022 alone, a quarter of respondents reported that their outages are costing over $1 million, while 45% reported their cost of outages are between $100,000 and $1 million. This marks a clear trend that cost of IT outages are steadily increasing, making investments in IT reliability more critical than ever [5].

According to Splunk, Global 2000 companies lose $400B annually due to application failures or slower. This includes direct financial losses from suspended operations and indirect losses like reputational damage, loosing customers [8]

In February 2017, Google released a report by analyzing over 900,000 mobile pages to assess mobile page speed performance across various industry sectors. The analysis revealed that for 70% of the pages examined, it took nearly 7 seconds for the visual content above the fold to display, and more than 10 seconds to fully load all visual content [9].

A significant portion of mobile pages were found to be excessively large, with 70% over 1MB, 36% over 2MB, and 12% exceeding 4MB. The study also indicated that as page load time increases from 1 second to 7 seconds, the probability of a mobile site visitor bouncing increases by 113%[9].

C. Preventive measures with performance testing

Addressing the causes of IT outages necessitates a need to understand the root causes of outages and the implementation of robust performance testing. The findings of effects of IT outages highlights the critical importance of performance testing in today's fast pacing digital world. Slower applications not only frustrate users but also impact organization financially. Having a proper test environment and regular performance testing helps identify root causes and reduce its effects by helping businesses enhance user engagement, reduce abandonment rates, and ultimately drive better financial outcomes.

Here are some of the strategies to adapt anticipated and unforeseen challenges that can achieved with performance testing.

- Execute performance tests early in development.
  - Steps to achieve it.
    - Have a clear expectations and success criteria such as acceptable response times.
    - Choose right test environment which is realistic and isolated from other Development, QA environments.
    - Simulate real world scenarios using performance testing tools like LoadRunner or JMeter.
    - Conduct early profiling in development stage to catch resource intensive code paths.
    - Enable continuous monitoring using application monioring tools like Dynatrace or Prometheus.
    - Perform regression testing for every code release.
  - Outcomes.
    - Identify potential performance issues before they occur in production affecting customers.
    - Reduce the chances of inefficient coding.
    - Prepare the teams for the unexpected.
- Monitor system utilization.
  - Steps to achieve it.
    - Instrument monitoring tools like Dynatrace, Grafana, Cloud Watch to monitor system performance metrics like CPU, memory, and disk utilization.
    - Enable continuous monitoring and visualize key performance metrics on dashboard for real time analysis.
  - Outcomes.
    - Detects performance anomalies and resource bottlenecks.
    - Reduces unplanned downtime through proactive monitoring.
    - Improves observability and reliability.
- Conduct different types of testing based on the load.
  - Steps to achieve it.
    - Identify load patterns in production.
    - Set performance benchmark goals based on business requirements.
    - Identify the type of test required like stress, endurance, spike, and negative-scenario tests.
    - Execute the load tests using performance testing tools.
    - Identify bottlenecks, optimize, and retest until goals are achieved.
  - Outcomes.
    - Identify weakest components.
    - Identify the causes of system crashes.
    - Identify configuration related issues that happen only under load.
- Execute Chaos testing.
  - Steps to achieve it.
    - Identify mission critical and vulnerable components.
    - Choose right tool like Gremlin to induce performance bottlenecks.
    - Plan for chaos experiments that align with real-world failure scenarios.
    - Enable continuous monitoring during the test to identify the problem pattern.
    - Fix vulnerabilities and revalidate the fixes through multiple tests.
  - Outcomes.
    - Identify hidden weaknesses in the system.
    - Ensures system remain resilient.
- Introduce disaster recovery testing.
  - Steps to achieve it.
    - Identify the mission-critical systems.
    - Choose disaster recovery test type like full interruption.
    - Create and activate the DR plan.
    - Monitor during disaster recovery, analyze the outcomes, and improve to reduce the gaps.
  - Outcomes.
    - Ensures that critical systems can be restored during crashes.
    - Create readiness during disasters.
    - Refines recovery strategies based on test results.
- Cloud auto scaling.
  - Steps to achieve it.
    - Define scaling strategy based on usage metrics like CPU > 70%.
    - Enable real-time performance metrics monitoring.
    - Configure auto scale policies in cloud platforms.
    - Test scaling up/down based on the usage and optimize thresholds.
  - Outcomes.
    - Increase uptime during traffic surges.
    - Uses infrastructure efficiently and reduces costs
    - Ensure seamless user experience during traffic surges.
- Automate testing process in continuous delivery.

- Steps to achieve it.
  - Integrate the process using automation tools to trigger tests automatically when build is triggered.
  - Trigger the tests when code deployment job is triggered.
  - Monitor the test results and improve the process based on the trends.
- Outcomes.
  - Performance degradation is detected automatically for every release.

V. Case Studies: The cost of downtime

Downtime of service unavailability can be due to maintenance or unexpected failures. Even a few minutes of downtime can lead to revenue loss and customer dissatisfaction. Here are some of the real world outages to understand the importance of robust performance testing in mitigating these outages [10].

A. Case Study 1: Azure Resource Manager exhausts capacity

Azure Resource Manager is the central tool that is used to deploy, manage and control Azure based resources.

1) Date of the incident: January 21, 2024.

2) Issue: Azure Resource Manager nodes failed on startup and more resources were consumed by the failed nodes, exhausting capacity.

3) Root cause: A configuration change gave preview access to new feature in June 2020 that has a code defect. This made nodes fail to startup.

4) Effect: Impacted downstream Azure services that rely on Azure Resource Manager to be unavailable.

5) Downtime: 7 hours.

6) Implications: A configuration change may not seem like affecting performance. In some cases, these changes can still have unexpected effects that impacts the performance which highlights the importance of performance testing in every stage of development.

7) Strategy to avoid this issue: Implementing negative scenarios as part of performance testing can help avoid these issues.

B. Case Study 2: Jira users seeing 503 service unavailable

Atlassian Jira is a tool that provides teams to plan and track work across different stages of the project.

1) Date of the incident: January 18, 2024.

2) Issue: Users of Atlassian Jira unable to track the status of their work as they saw 503 service unavailable errors.

3) Root cause: A scheduled database upgrade degraded the performance.

4) Effect: Caused an increase in back pressure which made requests to timeout.

5) Downtime: 3.5 hours.

6) Implications: Database upgrades require rigorous performance testing with higher load than expected due to potential changes in the database structure and indexing mechanisms. These changes, if not thoroughly tested can affect system performance and sometimes causes system outages.

7) Strategy to avoid this issue: Executing rigorous performance testing with all possible critical scenarios can help in avoiding these issues. This can be achieved with proper requirements gathering and test plan.

C. Case Study 3: Microsoft 365 outage

Microsoft 365 is a personal or business subscription service that provides services and apps for personal and business purposes.

1) Date of the incident: November 25, 2024.

2) Issue: Users saw 503 service unavailable errors while using Microsoft services.

3) Root cause: A change that surged number of requests being routed through servers, thereby affecting system performance.

4) Effect: Impacted processing capabilities of the infrastructure.

5) Downtime: It is not complete downtime but affected services for 7 hours.

6) Implications: A small change can lead to surge in incoming traffic, stressing the systems and causing service disruptions. This incident underscores the importance of load testing and continuous monitoring to detect bottlenecks from the traffic patterns. Although it is not complete downtime, even partial outages can significantly affect user experience.

7) Strategy to avoid this issue: Executing Spike testing as part of performance testing which replicates these sudden surge in requests can avoid these issues.

D. Case Study 4: Netflix broadcast disruptions

Netflix faced issues while broadcasting the live streaming of Jake Paul vs. Mike Tyson boxing event. Although this wasn't its first live streaming attempt, it was reported that its the most streamed event.

1) Date of the incident: December 20, 2024.

2) Issue: Netflix users reported that the service was not available a head of the live boxing event

3) Root cause: Netflix uses Open Access appliances (OCA) to store and deliver video content. These OCAs are preloaded with content during non-peak hours, while live streams happened in real time. These OCAs could not keep up with surge in traffic.

4) Effect: Received 500,000 reports that users were having problems streaming the match[11].

5) Downtime: 6 hours.

6) Implications: This disruption emphasize the importance of performance testing to ensure systems can handle peak load, quick response times and scale efficiently based on demand.

7) Strategy to avoid this issue: Measuring current production load and evaluating the performance of the system with 20% more than peak production volume.

E. Case Study 5: J. Crew website availability dropped

J. Crew is an American clothing retailer that sells clothing, shoes and accessories.

1) Date of the incident: November 23, 2018.

2) Issue: Shoppers were not able to make purchases and frequently bumped with "hang on a sec" message.

3) Root cause: Application servers couldn't keep up with the load.

4) Effect: J.Crew lost $775,000 due to unsold inventory

5) Downtime: 5 hours.

6) Implications: This case study emphasizes the importance of executing performance testing to prepare for peak-season to ensure the system is ready to handle peak load or scale based on the demand.

7) Strategy to avoid this issue: Environment setup to execute performance tests before peak season in regular intervals like holiday readiness tasks with increased load than previous year can help the systems perform the best during peak season.

VI. Conclusion and future work

In conclusion, since outages can occur anytime when we least expect, implementing thorough performance testing can help in minimizing the risk. By implementing performance testing early in the development, and helps to identify bottleneck early. Some of the outages can happen even with rigorous performance testing, the efforts mentioned in the paper can help identify the bottlenecks and solutions faster to ensure uninterrupted services to customers. Performance testing isn't just about avoiding downtime, it can ensure systems can perform flawlessly even under heavy load.

Future work will explore case studies that benefited from performance testing. More details on types of performance testing implemented in the case studies, and strategies to maintain system reliability to 99% will also be researched.

References

[1] E. Klotins, T. Gorschek, K. Sundelin, and R. Berntsson Svensson, "Towards cost-benefit evaluation for continuous software engineering activities.," *Empirical Software Engineering*, vol. 27, p. 157, 2022. DOI: 10.1007/s10664-022-10191-w.

[2] X. Han and T. Yu, "An empirical study on performance bugs for highly configurable software systems," ser. ESEM '16, New York, NY, USA: Association for Computing Machinery, 2016, ISBN: 9781450344272. DOI: 10.1145/2961111.2962602.

[3] M. R. Woodward and M. A. Hennell, "Strategic benefits of software test management: A case study," *Journal of Engineering and Technology Management*, vol. 22, no. 1, pp. 113–140, 2005, Research on Social Networks and the Organization of Research and Development, ISSN: 0923-4748. DOI: https://doi.org/10.1016/j.jengtecman.2004.11.006.

[4] S. Zaman, B. Adams, and A. E. Hassan, "A qualitative study on performance bugs," in *2012 9th IEEE Working Conference on Mining Software Repositories (MSR)*, 2012, pp. 199–208. DOI: 10.1109/MSR.2012.6224281.

[5] U. institute, "Annual outages analysis 2023," Last accessed: February, 2025, 2023, [Online]. Available: https://datacenter.uptimeinstitute.com/rs/711-RIA-145/images/AnnualOutageAnalysis2023.03092023.pdf.

[6] Magnitia, "Software testing statistics – 2023," Last accessed: February 20,2025, 2023, [Online]. Available: https://magnitia.com/blog/software-testing-statistics-2023.

[7] A. Petrosyan, "Most common root causes of it system and software-related outages worldwide," Last accessed: February, 2025, 2023, [Online]. Available: https://www.statista.com/statistics/1482105/it-system-software-related-outages-root-cause/.

[8] Splunk, "The hidden costs of downtime strike below the surface," Last accessed: February, 2025, 2024, [Online]. Available: https://www.splunk.com/en_us/campaigns/the-hidden-costs-of-downtime.html.

[9] Google, "Find out how you stack up to new industry benchmarks for mobile page speed," Last accessed: February, 2025, 2017, [Online]. Available: https://think.storage.googleapis.com/docs/mobile-page-speed-new-industry-benchmarks.pdf.

[10] C. thousand eyes - internt research team, "Internet and cloud intelligence blog," Last accessed: February, 2025, 2024, [Online]. Available: https://www.thousandeyes.com/blog/?cat=outage-analyses.

[11] J. Yoon, "Thousands report netflix livestream crashes during mike tyson-jake paul fight," Last accessed: February, 2025, 2024, [Online]. Available: https://www.nytimes.com/2024/11/16/business/media/netflix-outage-crash-boxing.html.