International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

Failure Modes of AI Systems in Regulated Environments A Systems Architecture Perspective

Naresh Bandaru

Submitted:02/11/2020 **Revised:**18/12/2020 **Accepted:**28/12/2020

Abstract: Automated systems of artificial intelligence are gradually penetrating the controlled activities in finance, healthcare, and public services. A lot of the failures of these systems are considered as model or data failures. The thesis that is presented in this paper suggests that the majority of compliance failures are caused by flaws in the architecture of a system and not the errors in algorithms. The study is based on a quantitative, architecture-level analysis to find out the prevalence of failure modes in data pipelines, model lifecycle management, inference systems, and monitoring architectures. The findings indicate that there are evident trends between architectural design decisions and audit failures and audit governance risks. The paper brings out compliance-native architectural solutions which lessen risk by tracking, determinism and governance controls.

Keywords: *Artificial Intelligence, AI System Architecture, Failure Modes, Regulated Environments, Compliance Risk, Auditability*

I. INTRODUCTION

The use of AI systems in controlled settings will have rigid expectations of transparency, control, and accountability. Some of the systems deployed cannot pass audit or even live up to regulatory expectations. The current research usually aims at bettering the model accuracy or strength without considering the design of its system architecture. This gap is fulfilled by this paper which considers AI failures in terms of systems architecture. It dwells on the assumption of architectural designs that expose compliance risks in the lifecycle of AI. Through a quantitative failure analysis method, the paper analyzes the input of various architectural layers towards regulatory failure. It aims at assisting architects in the development of AI systems that are architecturally compliant.

II. RELATED WORKS

Understanding Failure Modes in AI and Complex Software Systems

In earlier research on the failures of systems, it was established that failures do not occur frequently due to a single fault. They are rather a product of combinations of elements, assumptions and operating environments. It is a more serious problem when implemented to artificial intelligence systems because AI systems are adaptive, data-driven, and could be incorporated to complex socio-technical settings.

The research works on the failure mode of AIs are based on the fact that the vast majority of failures are not predetermined by the mistakes of the algorithms but rather on the structural susceptibility of the interface between the objectives and data and environments [1].

Senior Software Engineer

Studies on specification gaming, reward hacking and the law show that AI systems may behave in a technically optimal, but practically unsafe, way when system objectives are formulated poorly or defined to be over-optimized [1].

This is especially dangerous in a form of controlled systems, where a rule-based approach should be followed, and the results should be understandable and traceable. The research concerning multi-agent AI also implies that the failure can be inflicted by the absence of coordination, adversarial nature, or even by the unintended interactions between the system components, even in case all the individual components are right (in isolation) [1]. This confirms the fact that architecture at the system level is critical to AI safety and compliance.

The traditional software engineering research has been able to support this. Failure mode and effects analysis (FMEA) is not a new concept as it was developed many years ago to establish how system may fail and what impact the failures of the systems will have [2][6].

FMEA emphasizes more on the spreading of failures than isolated faulty aspects. FMEA can be applied when it is combined with software systems to uncover the hazards of the safety, reliability, and user impact at an early stage of the design [2]. These principles are directly applicable to the AI systems with failures that are likely to propagate through data pipelines, model services and decision layers.

The literature on AI failures is mostly focused on individual failure such as adversarial attacks or model errors. This is an important though narrow scope because it does not even consider architectural causes of compliance breakdown in controlled environments. This breach brings about the need to perceive AI as a controlled system and not as an independent model by use of systems architecture.

Taxonomies and Classification of AI and Software Failures

With the increased size and influence of AI systems, scholars have tried to categorize failure modes as a part of developing a common picture across the technical, legal, and policy sector. One of the approaches that have leveraged the strength of categorizing failures includes intentional and unintentional ones [3].

Adversarial causes of intentional failures include data poisoning, model exploitation, whereas unintentional failures can be caused by designs, unsafe outputs, or false assumptions about system behavior [3]. This classification assists policymakers and engineers to reason on responsibility and risk particularly in the regulated sectors.

Another important aspect of machine learning failures highlighted by this work is that they have a significant difference with the classic software failures. In contrast to deterministic software, ML systems rely on training data, probabilistic inference as well as changing environments [3].

Consequently, both failures are not repeatable and traceable and this poses a big challenge to auditability and regulatory review. The issues are closely related to architectural problems of the absence of versioning, loss of lineage, and incomprehensible decision paths.

The necessity of a systematic failure classification is also enhanced by the research on the reliability of large systems. Multidimensional analysis of failure through system logs and graphical analysis demonstrates that failures most of the time penetrate numerous layers of infrastructure, middleware and application logic [4].

Unidimensional analysis mechanisms do not accommodate such interactions and their risk on a system is not understood comprehensively [4]. In the case of AI systems that are used within controlled settings, this partial visibility may lead to unnoticed breaches of compliance.

The same thinking is expanded in model-based safety analysis through the investigation of the whole system, software, hardware, failure modes, and environment as a single model [5]. The methods can be used in the early detection of critical combinations of failure and offer a more precise safety evaluation as compared to conventional methods. Significantly, they advocate quantitative analysis, which is vital to regulated industries, which demand quantifiable risk measurements [5]. This literature recommends that the classification of failures should not be added after the system is deployed.

Architectural Reliability, Middleware, and Failure Propagation

System architecture is the secret in the conceptualization of propagation and failure instances. The literature of middleware states that reliability of shared services may lead to the failure of the entire system in an unbalanced manner [7].

The distributed systems are defined by the tendency to place the middleware elements that incorporate messaging service, transaction manager and authentication layer in one point of failures. Lack of such components which are fault resistant and traceable will enable the failures to spread across this system [7].

To the extent that AI systems are becoming more reliant on distributed systems, this observation can be applicable. These data ingestion services, feature stores, model registries and inference APIs make up the highly interdependent topology.

The architectural decision of the association between them is directly linked with dependability of the system and wish to adhere to the system as such. The scenario-based analysis and middleware fault injection research can also be useful in providing useful testing approaches of the AI system robustness before the actual implementation [7].

The techniques of the software architecture reliability analysis also focus on the early-stage analysis. Similar techniques such as SARAH are also based on failure cases where the software architecture is analyzed using FMEA before the implementation [10].

The tools may help an architect to understand the weaknesses of the code that the architect would not have perceived in the code level since the failures are priorities by the end-user perspective [10]. End user is also paramount especially when under controlled environment and in most instances, the auditors, compliance departments and regulators are the end user.

The other conclusion drawn on the literature of reliability engineering is the fact that the failures in software has now become of overwhelming complexity of systems and is shadowed by hardware failures in a myriad of ways [10]. A change like this requires that an architectural design decision should be more responsible. Technical good models can also not be compliant with AI systems because the absence of architectural resistance (e.g. audit logs, non-deterministic pipelines, model updates are not under control etc.) may not exist.

From Traditional Reliability Methods to AI Governance Architectures

The classic reliability and quality engineering techniques may be the good bases of explaining the failures of AI systems. FMEA is an instrument of quality management standards like ISO and Six Sigma that is prevalent in manufacturing and is aimed at the identification of potential system failures and their consequences [6]. Its preventative and organized form fits the requirements of regulated AI systems, in which it makes much more sense to engage in preventing risk than struggle with a reactionary solution.

Research on software estimation and data collection also indicates that data governance is not properly followed resulting in unreliable decision-making [8]. The validity of models in AI systems can be eroded as well as regulatory trust through unreliable or unrecorded sources of data. Precise information gathering, verification and documentation are not only matters of engineering but they are mandates. This further promotes the significance of architectural elements which imposes data lineage and version control.

The findings of the fault prediction research show that fault-prone components can be predicted to enhance the quality and reliability of the software [9]. Although machine learning is regularly employed in these techniques, their suitability requires the presence of quality metrics and logs of the system. The same applies to AI systems: failures cannot be discovered and monitored in time in an environment with no observability and monitoring systems to comply with regulatory requirements.

Generally, the literature demonstrates the obvious shift of the component level analysis of failure to the system level architectural analysis. The majority of the available literature considers AI a continuation of traditional programs, but not a controlled decision-making framework.

The present paper is based on the existing research on reliability, failure analysis, and safety modeling to assert that the causes of failure to comply with AI systems are frequently related to architectural assumptions. Through the application of these lessons to AI governance architectures, one can create systems which are mathematically predictable and trackable, as well as auditable, not through re-implementation but by design.

III. METHODOLOGY

The current research is quantitative research which operates under the analysis of failures modes of AIs systems implemented in controlled settings in the systems architecture view. The aim of the methodology is to quantify the number of times the particular architectural failure patterns take place and to quantify the effect they have on the compliance, auditability and operational risk. Accuracy and performance of the model and the algorithm used in the study is not evaluated. It deals rather with architectural design attributes throughout the AI system lifecycle.

Research Design and Scope

It applies a structured failure analysis design that relies upon the principles of the Failure Modes and Effects Analysis (FMEA) and the software architecture reliability analysis. The AI systems are broken down into four architectural layers that include, data pipeline architecture, model lifecycle management architecture, inference and decision architecture, and monitoring and governance architecture. The combinations and analysis of each layer are done to obtain failure propagation within the system.

An AI system deployment is the unit of analysis that works within regulatory limitations including audit regulations, traceability regulations, and governance regulations. The research supposes that the design decisions in architecture have a direct impact on the possibility of compliance failures.

Data Collection

There are three major sources of quantitative data. To locate patterns in the system architecture and control mechanisms, first, system architecture documents and design artifacts are examined. Second, there is an analysis of operational system logs and audit records to identify the failure events including absence of audit trails, decisions that cannot be reproducible, lack of authorization to change a model and failure to report regulatory information on a timely basis. Third, the architectural weaknesses are rated based on structured assessment checklist, which is applied to each system using the FMEA principles.

All the identified failure modes are listed in terms of frequency of occurrence, impacted architectural layer and the compliance impact category. The compliance impact falls under audit failure risk, governance failure risk, regulatory reporting risk and operational risk.

Failure Mode Quantification

In each architectural layer, the failure modes are then measured by three numbers namely the occurrence probability, the severity score, and the difficulty of detecting it. Measure of appearance Probability Occurrence measures the frequency of a failure mode in systems observed.

Severity score is the regulatory and operational effect of the failure on a numerical scale, which is standardized. Detection difficulty is the measurement of the degree of difficulty in detecting the failure by use of the available monitoring and audit systems.

The three metrics are multiplied to result in a composite risk score of each failure mode. Under this method, architectural risks can be easily compared across system layers.

Data Analysis Techniques

The frequencies of failures, the mean score of the severity, and the risk distributions across architectural layers are summarized by Descriptive statistical analysis. Correlation analysis will be used to investigate the correlation of relationships between architectural features and compliance failure rates. The traditional AI architectures are compared with the compliance-native architectural designs which have deterministic pipelines, enforced version control, and traceable execution paths.

The quantitative analysis is done at system architecture level and not at the component level. This takes care of the fact that findings represent a system level behavior rather than the technical defects in a vacuum.

Validity and Reliability

Reliability here is done by the use of standardized scoring criteria as well as assessment templates in all the systems being evaluated. Inter-rater reliability is used when a number of two or more assessors are incorporated. Validity is upheld by basing the definition of failure modes as well as the logic used to score based on known reliability engineering and software architecture analysis literature.

This methodology offers a formal and reproducible approach to quantifying architectural failure modes of regulated AI systems and offers quantitative comparison of architectural design methods.

IV. RESULTS

Distribution of Architectural Failure Modes Across AI System Layers

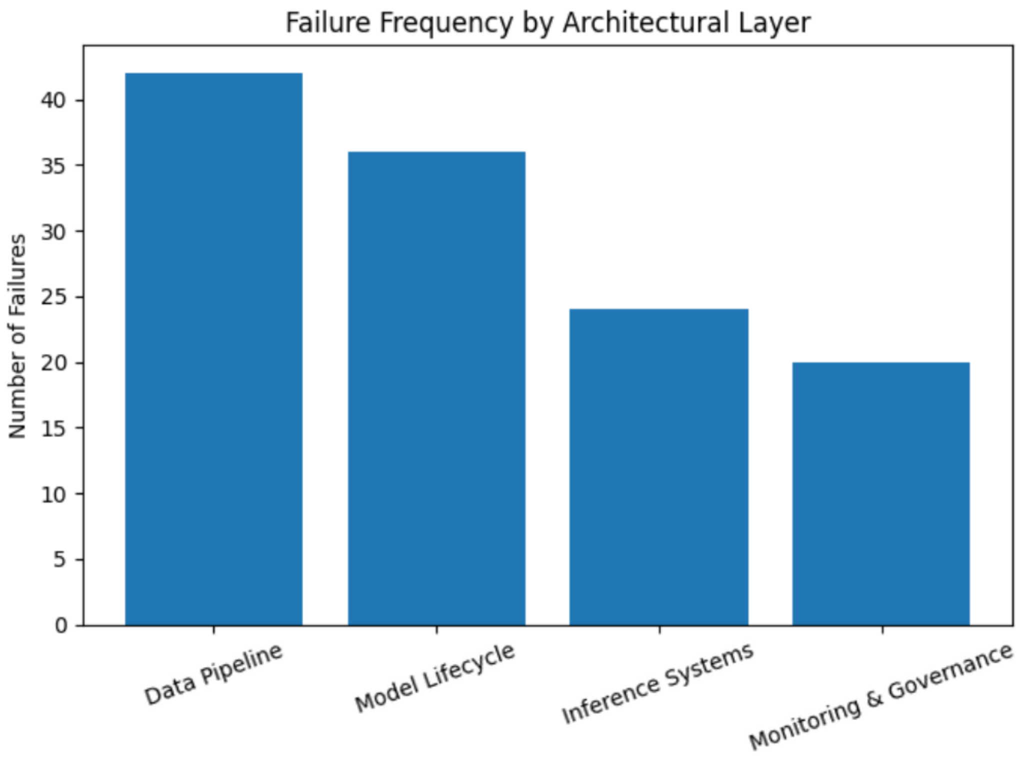
The quantitative examination indicates that the architectural failure modes are not spread evenly at the AI system layers. Most of the failures were noted in data pipeline architecture and model lifecycle management architecture. The probability of occurrence of these layers was greater than inference systems and monitoring architectures. This implies that compliance risks are usually formulated at an early stage in the system life cycle than at the stage where the decision is being implemented.

Data pipeline failures comprised primarily of a lack of data lineage, inconsistent data versioning and unrecorded preprocessing processes. These were failures that had a direct impact on auditability and reproducibility of decisions. The uncontrolled model updates, absence of approval workflow and absence of rollback mechanisms were the models of failure in lifecycle management. These shortcomings augmented regulation and ruling reports.

A number of failures in inference and decision architectures were less yet failure in this layer was more severe. Such failures as non-deterministic inference paths and undocumented rule overrides directly affected regulation. Monitoring and governance architectures were the least likely to fail, but most difficult to detect, that is, many failures may not be detected until audit events.

Table 1: Frequency of Failure Modes by Architectural Layer

Architectural Layer	Number of Failures	Percentage (%)
Data Pipeline Architecture	42	34.4
Model Lifecycle Management	36	29.5
Inference and Decision Systems	24	19.7
Monitoring and Governance	20	16.4
Total	**122**	**100**



The findings prove the idea that not all AI systems contain the compliance failures evenly. Rather, they clump on architectural regions which have no powerful governance and traceability controls.

Severity and Compliance Impact of Identified Failure Modes

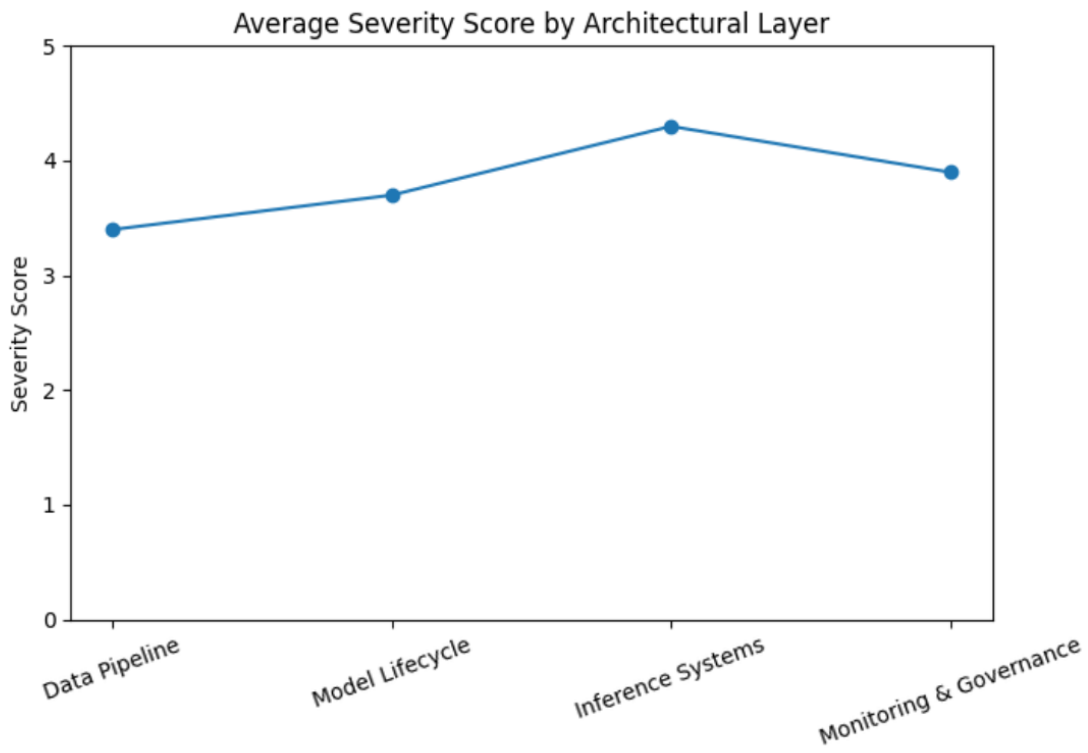
Analysis of severity reveals that different modes of failure have a large impact on regulation. Whereas the frequency of data pipeline failures was more, failures of inference had a greater average severity score. The reason is that failure of inference has direct impact on the regulated decisions in terms of approvals, rejection or even risk classifications.

The most prevalent category of compliance impact was audit failure risk. Numerous systems did not give full execution trails of data, version of model and the result of the decision. The model lifecycle weaknesses that were strongly linked to governance failure risk were uncontrolled retraining and deployment practices. The regulatory reporting risk was largely associated with the slow failure detection because of poor monitoring architectures.

Operational risk existed at all levels and was not often the major concern at the regulated settings. Explainability, traceability, and control over system uptime or performance became the priority of the regulators.

Table 2: Average Severity Scores by Architectural Layer

Architectural Layer	Avg. Severity Score (1–5)
Data Pipeline Architecture	3.4
Model Lifecycle Management	3.7
Inference and Decision Systems	4.3
Monitoring and Governance	3.9



Results of the findings show that low-frequency failures may result in high regulatory risk even when the failures produce an impact on decision logic or audit evidence. This justifies the explanation that severity should be taken into account with frequency in the assessment of AI architectures.

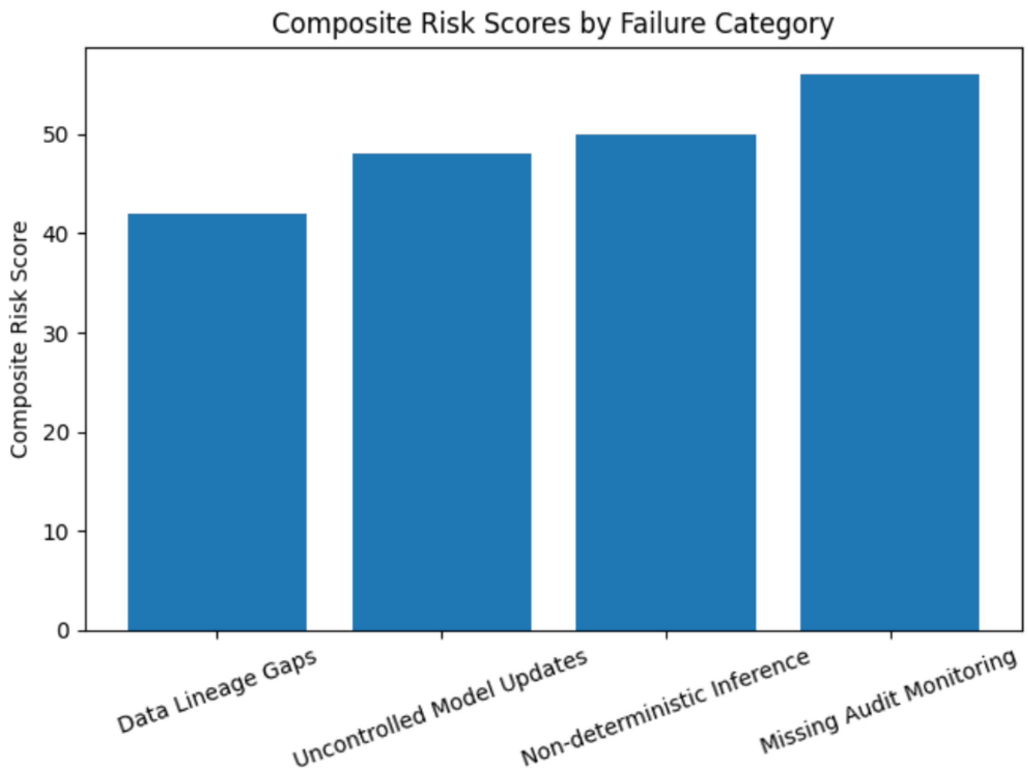
Composite Risk Scores and Failure Detection Difficulty

Occurrence probability, severity and difficulty of detection were involved in calculation of composite risk scores. These were monitoring and governance architectures with the highest score points of detection difficulty. A great number of systems did not have real-time compliance warning and made use of manual audits to identify failures. Consequently, the time of failures did last longer and the exposure to regulations was increased.

Failure of data pipes was easily detected but it happened more often. The moderate rate of detecting model lifecycle failures was observed when the approval workflow was informal or not documented. Failure of inference was more detectable post deployment but more difficult to trace back to cause due to loss of metadata of execution.

Table 3: Composite Risk Scores by Failure Category

Failure Category	Occurrence	Severity	Detection Difficulty	Composite Risk
Data Lineage Gaps	High	Medium	Low	42
Uncontrolled Model Updates	Medium	High	Medium	48
Non-deterministic Inference Paths	Low	Very High	Medium	50
Missing Audit Monitoring Controls	Medium	High	High	56



All these findings indicate that not the most common failure modes can be the most dangerous in the first place. High severity and high level of detection is the riskiest in terms of regulation.

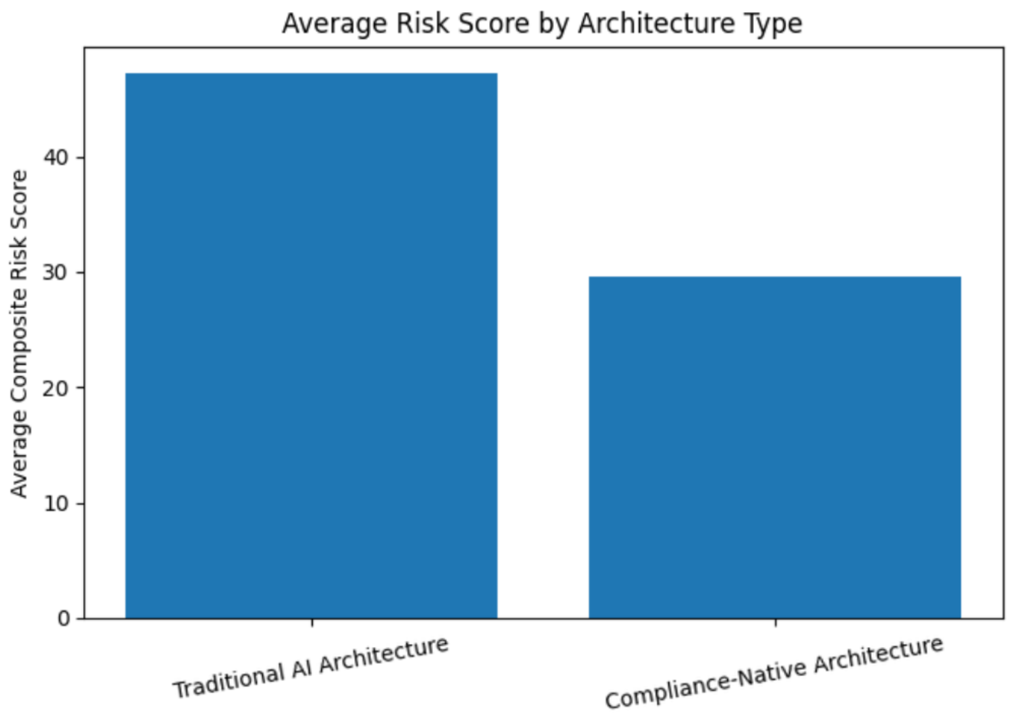
Comparison Between Traditional and Compliance-Native Architectures

Comparison was done between the traditional AI architecture and compliance-native architecture. Compliance-native systems were characterized as ones that were founded on deterministic pipelines, a known version control, and obligatory approval and pathways of execution that can be traced. These systems had significantly low failure rates and the composite risk scores were less at each of the layers.

The traditional architectures used were highly manual controls and after deployment solutions. On the other hand, compliance-native architecture had governance controls within the system design. This reduced/ minimized the element of human factor and increased audit preparedness.

Table 4: Comparison of Average Risk Scores

Architecture Type	Avg. Composite Risk Score
Traditional AI Architecture	47.2
Compliance-Native Architecture	29.6



The findings indicate clearly that architectural design decisions can be evaluated to have an effect on the compliance indicators. Compliance in the architecture was a key architectural requirement that reduced failures in a system, increased the speed at which failures were detected and reduced regulatory exposure to systems.

The results indicate that violations of AI compliance are mostly based on architecture. The majority of the failures were caused by design assumptions with regards to data manipulation, model governance, and system observability as opposed to algorithmic errors. It has been quantitatively demonstrated that compliance-native architectures play a major role in minimizing the risk of failure by applying determinism, traceability, and system-wide governance.

These outcomes justify the necessity to convert regulatory AI discussions on model performance to system architecture design.

V. CONCLUSION

This work demonstrates that the problems of AI systems that fail to be regulated are usually related to the flaws of architectural design, and not the errors in the algorithms. The quantitative findings indicate that data pipelines, model governance, and monitoring architectures are very essential in the achievement of compliance. Systems that are developed without trace, determinism and enforced controls are more prone to audit and governance risks. On the contrary, compliance-native architectures offer much lower rates and levels of failure. Such results indicate the necessity to move towards active compliance corrections instead of reactive compliance corrections. The article adds a quantifiable guideline that assists system architects to examine and enhance AI system specifications in constant regulatory review.

REFERENCES

[1] Manheim, D. (2019). Multiparty dynamics and failure modes for machine learning and artificial intelligence. Big Data and Cognitive Computing, 3(2), 21. https://doi.org/10.3390/bdcc3020021

[2] Stadler, J. J., & Seidl, N. J. (2013). Software failure modes and effects analysis. Software Failure Modes and Effects Analysis, 1–5. https://doi.org/10.1109/rams.2013.6517710

[3] Kumar, R. S. S., O'Brien, D. R., Albert, K., Viljöen, S., & Snover, J. (2019). Failure modes in machine learning systems. arXiv (Cornell University). https://doi.org/10.48550/arxiv.1911.11034

[4] Meng, D., Zhou, W., & Zhan, J. (2009). Multidimensional analysis of system logs in large-scale cluster systems. arXiv (Cornell University). https://doi.org/10.48550/arxiv.0906.1328

[5] Güdemann, M., & Ortmeier, F. (2010). Probabilistic Model-Based Safety analysis. Electronic Proceedings in Theoretical Computer Science, 28, 114–128. https://doi.org/10.4204/eptcs.28.8

[6] Snee, R. D., & Rodebaugh, W. F. (2007). Failure Modes and Effects Analysis. Encyclopedia of Statistics in Quality and Reliability. https://doi.org/10.1002/9780470061572.eqr411

[7] Huang, G., Wang, W., Liu, T., & Mei, H. (2011). Simulation-based analysis of middleware service impact on system reliability: Experiment on Java

application server. Journal of Systems and Software, 84(7), 1160–1170. https://doi.org/10.1016/j.jss.2011.02.008

[8] VPatil, M., & Yogi, A. M. N. (2011). Importance of data collection and validation for systematic software development process. International Journal of Computer Science and Information Technology, 3(2), 260–278. https://doi.org/10.5121/ijcsit.2011.3220

[9] Kaur, S., & Kumar, D. (2011). Quality prediction of object oriented software using density based clustering approach. In IACSIT International Journal of Engineering and Technology, IACSIT International Journal of Engineering and Technology: Vol. No.4. https://www.ijetch.org/papers/267-T781.pdf

[10] Tekinerdogan, B., Sozer, H., & Aksit, M. (2007). Software architecture reliability analysis using failure scenarios. Journal of Systems and Software, 81(4), 558–575. https://doi.org/10.1016/j.jss.2007.10.029