

# **AI Validation Systems: A Missing Architectural Layer for Reliable AI**

**Rishi Sood**

ORCID: 0009-0008-6479-4061

**Affiliation:** Independent Research Collaboration

**Corresponding Author:** [rishisood@protonmail.com](mailto:rishisood@protonmail.com)

**Date:** May 2026

**Keywords:**

AI system validation; AI reliability; validation systems; AI architecture; output validation; AI system design; human–AI interaction; AI governance; structured judgement; decision-making systems; iterative refinement

**License:**

Creative Commons Attribution 4.0 International (CC BY 4.0)

## **Abstract**

Current AI systems are highly effective at generating fluent and contextually relevant outputs, yet they lack a systematic mechanism for determining whether those outputs should be trusted. As a result, improvements in model capability and orchestration have increased what these systems can produce without proportionally improving their reliability.

This paper argues that the core limitation of modern AI systems is the absence of a dedicated validation layer. While existing approaches focus on enhancing generation through scaling, prompting, and agent-based orchestration, they do not provide a structured means of evaluating outputs before use. This creates a fundamental gap in system architecture: outputs are produced but not internally judged.

We introduce validation as a first-class architectural component, defined as a structured, adversarial process that evaluates generated outputs against objectives, constraints, and potential failure conditions. Unlike conventional evaluation methods, validation is embedded within the system, operates on each output, and directly determines whether outputs are accepted, revised, or rejected.

We formalise validation as a mechanism consisting of objective anchoring, adversarial evaluation, structured judgement, and decision output. Integrated into a system pipeline, validation functions as a control point that governs output propagation and enables iterative refinement through repair.

We argue that reliability in AI systems cannot be achieved solely through improvements in generation, but instead emerges from structured validation that transforms hidden errors into observable risks. This reframes AI system design from generation-centric to validation-centric architectures, with implications for system reliability, human–AI interaction, and the development of trustworthy AI systems.

## **Section 1 — Introduction**

Large language models have introduced a new class of systems capable of generating fluent, coherent, and contextually relevant outputs across a wide range of tasks. However, fluency has often been mistaken for reliability. In practice, these systems frequently produce outputs that are plausible but incorrect, incomplete, or misaligned with real-world constraints (Ji et al., 2023).

This creates a fundamental trust problem. Users are presented with outputs that appear authoritative, yet lack any internal mechanism for determining whether they should be trusted. As a result, the burden of evaluation is shifted to the human user, who must continuously assess correctness, identify hidden assumptions, and verify applicability before acting on the output. Over time, this leads to cognitive overload, reduced trust, and unreliable human–AI interaction (Glikson & Woolley, 2020).

The core limitation is not that AI systems generate imperfect outputs, but that they generate outputs without a built-in process for judging their reliability. Current systems are architecturally designed to produce responses, but not to determine whether those responses are valid, consistent, or safe to use. As outputs become more fluent and complex, this absence of internal judgement becomes increasingly problematic.

Existing approaches to improving AI performance focus on increasing model capability, refining prompting techniques, and developing more sophisticated orchestration frameworks. While these approaches expand what systems can do, they do not address whether the outputs produced should be accepted. As a result, capability improves without a corresponding improvement in reliability.

This paper argues that the central limitation of current AI systems is the absence of a validation layer: a structured, internal mechanism that evaluates generated outputs before they are used. We define validation as an adversarial, systematic process that detects error, misalignment, and failure risk by assessing outputs against objectives, constraints, and potential real-world conditions.

AI systems cannot guarantee correctness due to their probabilistic nature. However, they can become reliably usable if they include a structured validation layer that systematically detects and exposes potential failure before outputs are used.

The validation mechanisms described in this paper reflect patterns that have emerged through the development and analysis of governed AI systems, including documented system behaviour in structured interaction records (Sood, 2025a–2026c; Mahdi, 2025).

This work builds on a broader research program examining the reliability of AI systems through governance and structured interaction. To situate the present contribution, we briefly outline this lineage and its progression toward the current focus on validation.

## **Research Lineage**

This work builds on a series of prior papers that examined the reliability of AI systems through the lens of governed, distributed cognition. Earlier work established that reliability does not emerge solely from model capability, but from structured system-level processes that manage drift, enforce constraints, and enable iterative repair (Sood, 2025a, 2025b, 2025c, 2025d, 2026a, 2026b, 2026c).

These systems were characterised by controlled execution pipelines in which outputs are generated, evaluated, and revised across successive stages (Sood, 2025a, 2025b, 2025c). Within this framework, validation was implicitly present as part of governance, particularly in stages responsible for verification and correction.

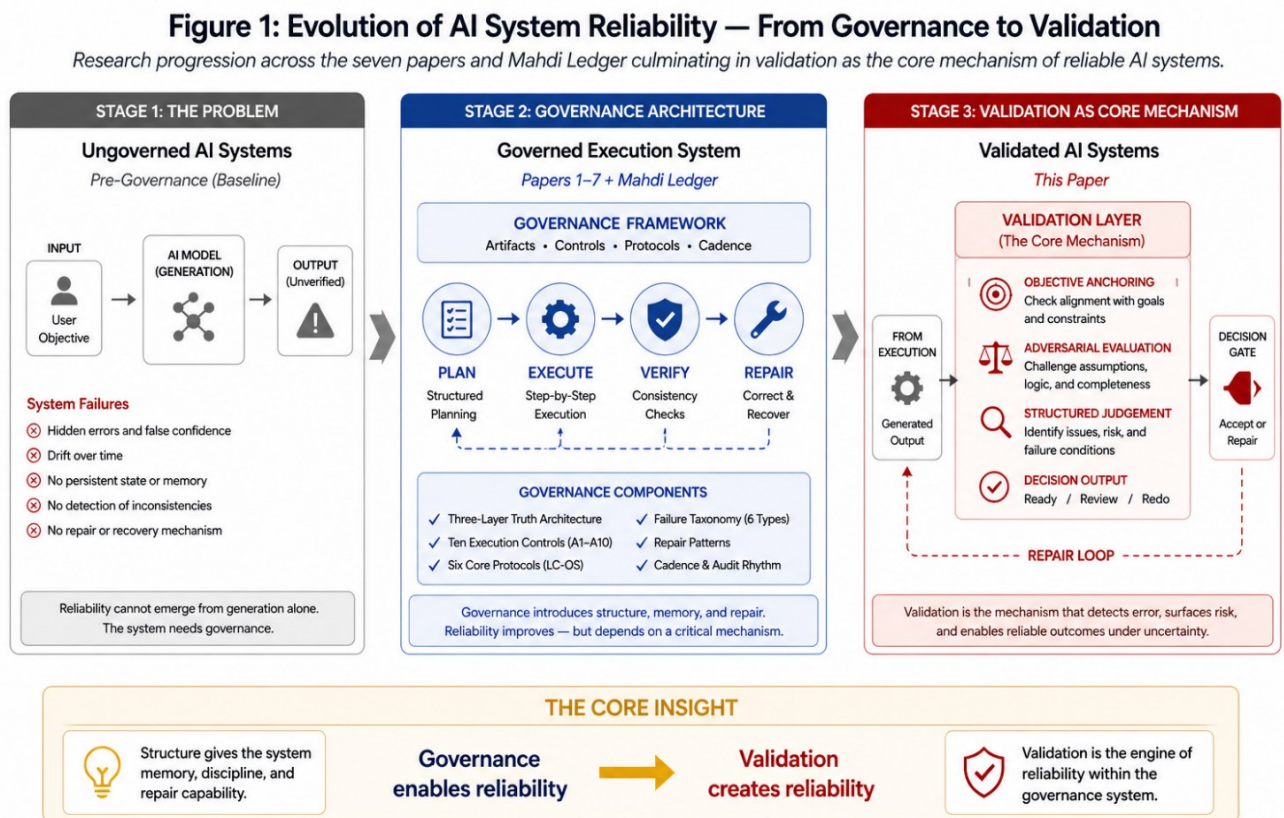
However, validation was not previously isolated as an independent system function. The present work extends this research by extracting validation from the broader governance architecture and formalising it as a distinct layer within AI systems.

This represents a refinement rather than a departure. While earlier work demonstrated how governance enables stability and reliability, this paper clarifies the underlying mechanism through which that reliability is achieved. By positioning validation as a first-class

component, this work provides a more precise account of how AI systems detect, surface, and respond to potential failure.

## Research Progression

The body of work presented in this research program has evolved through three connected stages. Initial work identified the reliability gaps in AI systems. Subsequent work introduced the governance architecture required to manage those gaps. The present work isolates validation as the core mechanism within governance that enables reliable system behaviour under uncertainty. **Figure 1** summarises this progression.



This figure illustrates the progression of the research program. Initial work identified the reliability problem in AI systems (Stage 1). Subsequent work built governance frameworks to manage this problem (Stage 2). This paper isolates validation as the core mechanism within governance that ultimately enables reliable AI system behaviour (Stage 3).

## Figure 1 — Evolution of AI System Reliability — From Governance to Validation

*This figure illustrates the progression of the research program. Initial work identified the reliability problem in AI systems (Stage 1). Subsequent work built governance frameworks to manage this problem (Stage 2). The present paper isolates validation as the core mechanism within governance that ultimately enables reliable AI system behaviour (Stage 3). While earlier work established governance as a means of controlling system behaviour, the present work clarifies that validation functions as the central process through which error is detected, risk is surfaced, and reliability is achieved under conditions of uncertainty.*

## Section 2 — Limits of Current Approaches

Efforts to improve the reliability of AI systems have primarily focused on increasing model capability, refining prompting techniques, and developing more sophisticated orchestration frameworks. While these approaches have significantly expanded what AI systems can do, they do not provide a mechanism for determining whether the outputs produced are reliable. As a result, systems have become more capable without becoming proportionally more trustworthy.

One major line of improvement has been model scaling. Larger models trained on more data demonstrate improved performance across a wide range of tasks, including reasoning, language understanding, and code generation. However, scaling does not eliminate error (Bender et al., 2021). Because these systems remain probabilistic in nature, they continue to produce incorrect or misleading outputs alongside correct ones. Although scaling can reduce the frequency of failure in some cases, it does not provide any mechanism for identifying when a specific output is wrong or should not be trusted.

A second approach focuses on prompting and alignment techniques. Carefully designed prompts and instruction strategies can guide models toward more useful and contextually appropriate outputs. These techniques can improve consistency and reduce certain classes of errors. However, they remain inherently fragile. Small changes in phrasing or context can lead to significantly different outputs, and the effectiveness of prompting depends heavily on user skill. More importantly, prompting influences how outputs are generated but does not introduce any structured process for evaluating whether those outputs are reliable after generation. Existing evaluation approaches, such as benchmarks, remain external and do not capture real-world reliability (Srivastava et al., 2022).

A third approach involves the use of agent frameworks and orchestration systems (e.g., Yao et al., 2022). These systems structure multi-step workflows, enable tool use, and coordinate complex tasks across multiple components. By introducing planning, memory, and execution layers, they improve the ability of AI systems to perform extended operations. However, their primary focus is on execution rather than evaluation. They enable systems to generate more complex outputs but do not inherently determine whether those outputs are valid, coherent, or aligned with real-world constraints.

Across all three approaches, a consistent pattern emerges. Improvements are directed toward producing better outputs, but not toward judging those outputs before they are used. As a result, the fundamental problem remains unresolved: AI systems can generate increasingly sophisticated responses, yet they still lack a built-in mechanism for determining whether those responses should be accepted, revised, or rejected.

### Section 3 — The Missing Validation Layer

The preceding analysis reveals a consistent structural limitation across current AI systems. While significant progress has been made in improving the ability to generate outputs, there remains no built-in mechanism for determining whether those outputs should be trusted. This is not a limitation of capability, but of system design.

At present, AI systems are architecturally configured to produce responses, but not to evaluate them. Generation is treated as the primary function, while judgement is either externalised to the user or handled informally through ad hoc processes. As a result, the system is incomplete: it can generate increasingly sophisticated outputs, yet lacks the means to determine whether those outputs are valid, coherent, or appropriate for use.

This gap becomes more pronounced as system capability increases. More fluent and complex outputs create a stronger appearance of correctness, while simultaneously making errors more difficult to detect. In the absence of an internal evaluation mechanism, reliability does not scale with capability. Instead, the burden of judgement is transferred to the user, who must compensate for the system's lack of internal scrutiny.

We argue that this limitation arises from the absence of a dedicated validation layer within AI system architecture. A validation layer is a distinct component responsible for evaluating generated outputs against their intended objectives, relevant constraints, and potential failure conditions before those outputs are accepted for use. Unlike generation mechanisms, which aim to produce plausible responses, validation operates by interrogating those responses, identifying weaknesses, and surfacing risks.

Existing approaches to evaluating AI outputs typically treat evaluation as external, optional, or retrospective. By contrast, validation as defined here is embedded within the system, operates on every output, and directly determines whether that output is accepted for use. This distinction shifts validation from a supporting technique to a governing system function that controls how outputs propagate through the system.

The introduction of validation as a first-class architectural component addresses this gap directly. Rather than treating evaluation as an external or optional process, validation embeds structured scrutiny within the system itself. Outputs are no longer assumed to be usable upon generation, but must first pass through a process that determines whether they should be accepted, revised, or rejected.

In this sense, the absence of validation is not a minor deficiency, but a fundamental architectural omission. Without a mechanism for internal judgement, improvements in generation will continue to increase capability without resolving the problem of reliability.

Without validation, outputs remain unqualified and cannot be reliably used. AI systems therefore operate as generators of unqualified outputs rather than reliable decision-support systems. Validation is not an enhancement, but a necessary condition for system reliability.

## Section 4 — Defining Validation Systems

Having established the absence of a validation layer as a structural limitation, we now define what such a layer entails. A validation system is not an auxiliary feature or post-hoc check, but a distinct system function responsible for determining whether generated outputs are suitable for use.

We define an AI validation system as a structured process that evaluates generated outputs against their intended objective, relevant constraints, and potential real-world conditions to detect error, misalignment, and failure risk before those outputs are accepted.

Validation differs fundamentally from generation. While generation mechanisms aim to produce plausible and contextually appropriate responses, validation systems are designed to interrogate those responses. Their purpose is not to improve fluency or optimise surface quality, but to assess whether an output meets the conditions required for reliable use.

Validation also differs from static evaluation methods such as benchmarks or offline testing. Rather than operating externally or retrospectively, validation is embedded within the execution flow of the system and operates dynamically on each output. It is therefore context-sensitive, objective-driven, and directly coupled to system behaviour.

Unlike conventional evaluation methods, which provide descriptive assessments of performance, validation produces actionable decisions within the system. It determines whether outputs propagate forward, require revision, or are rejected, thereby directly influencing system behaviour rather than merely describing it.

Crucially, validation does not attempt to establish absolute correctness. Given the probabilistic nature of AI systems, correctness cannot be guaranteed. Instead, validation provides a structured method for identifying and assessing potential failure under uncertainty. It transforms implicit risks within an output into explicit, observable issues that can be examined and acted upon.

A validation system can be characterised by four core components. First, objective anchoring ensures that evaluation is grounded in the specific goal the output is intended to achieve. Second, adversarial evaluation focuses on identifying weaknesses by actively searching for hidden assumptions, missing constraints, and potential failure modes. Third, structured judgement organises identified issues according to their severity and impact, distinguishing between critical flaws and less significant concerns. Fourth, decision output translates this assessment into an actionable outcome, determining whether the output should be accepted, revised, or rejected.

Together, these components establish validation as a necessary system function. Rather than treating outputs as inherently acceptable, validation requires outputs to pass structured scrutiny before they are used, thereby shifting reliability from an assumed property to an assessed condition.

## Section 5 — Mechanism of Validation

Validation operates as a structured judgement system that determines whether generated outputs can be accepted, revised, or rejected. Rather than functioning as an auxiliary check, it serves as the mechanism through which system-level decisions about output usability are made. It transforms generated outputs into assessed outcomes, enabling the system to regulate its own behaviour under conditions of uncertainty.

The validation mechanism consists of four interrelated components.

**First, objective anchoring.** Validation begins by explicitly grounding the evaluation in the intended objective of the output. This ensures that assessment is not performed against generic notions of quality, but against the specific goal the output is meant to achieve. By maintaining a consistent reference to the objective, this component prevents drift and ensures that subsequent evaluation remains contextually relevant.

**Second, adversarial evaluation.** Rather than attempting to confirm correctness, the system actively searches for weaknesses within the output. This involves identifying hidden assumptions, detecting missing constraints, examining the depth and coherence of reasoning, and exploring potential failure modes under realistic conditions. The emphasis is on uncovering where and how the output could fail, rather than justifying why it appears valid. This adversarial stance is central to transforming validation from passive checking into active scrutiny. Mechanically, adversarial evaluation involves systematically interrogating outputs by generating counterfactual scenarios, identifying unstated assumptions, and testing the output against conditions under which it could fail. Rather than confirming correctness, the process is structured to surface potential breakdown points and expose hidden weaknesses in reasoning or applicability.

**Third, structured judgement.** The issues identified during adversarial evaluation are organised according to their severity and impact. Critical issues represent fundamental flaws that compromise the validity or usability of the output, while non-critical issues indicate areas of uncertainty or limited robustness. This categorisation enables the system to move beyond unstructured critique and produce interpretable assessments that can inform decision-making.

**Fourth, decision output.** The validation process culminates in a clear and actionable judgement regarding the output. Rather than presenting a list of observations alone, the system determines whether the output is suitable for use, requires revision, or should be rejected. This step transforms validation from analysis into system behaviour, enabling outputs to be filtered, revised, or iteratively improved based on identified issues.

Together, these components transform validation from an informal judgement into a system-level function that governs output acceptance.

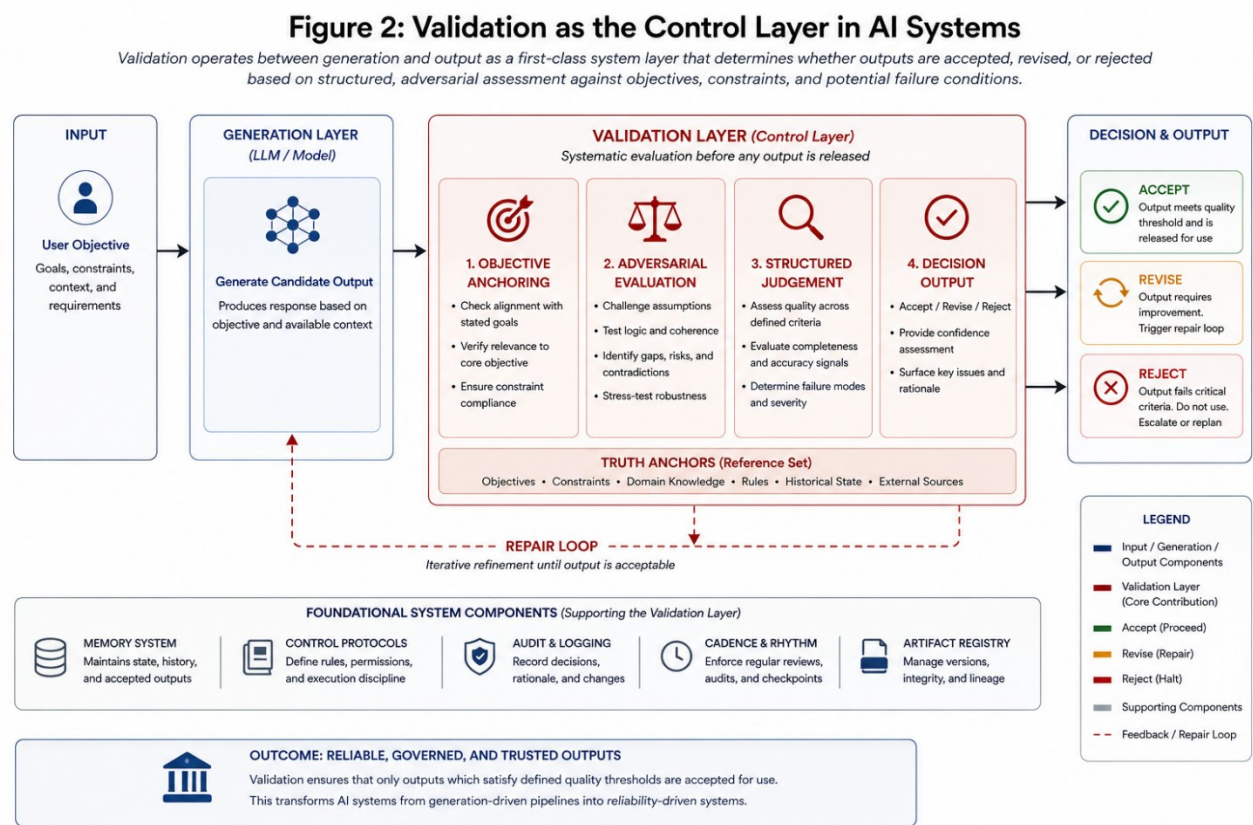


Section 6 — Validation in System Architecture

Having established validation as a distinct mechanism, we now consider its role within the broader architecture of AI systems. Validation is not an isolated process, but a functional layer that operates within the system’s execution flow, shaping how outputs are accepted, revised, or rejected.

A validation-enabled system can be understood as a sequence of interdependent stages. At a high level, this can be represented as: input, generation, validation, and output, with an optional repair stage following validation. In this structure, the generation component produces a candidate output, while the validation layer evaluates that output before it is presented for use. This introduces a critical separation between production and acceptance, ensuring that outputs are not treated as valid by default.

To illustrate how validation operates within system architecture, **Figure 2** presents validation as a control layer that governs whether generated outputs are accepted, revised, or rejected before use.



**Figure 2. Validation as the Control Layer in AI Systems**

*This figure illustrates validation as an embedded control layer positioned between generation and output. Generated outputs are not accepted by default, but are evaluated through a*

*structured validation process that produces explicit decisions: accept, revise, or reject. These decisions determine whether outputs are used directly, returned for repair, or discarded. By controlling output propagation and enabling iterative refinement, validation transforms AI systems from generation-driven pipelines into decision-regulated systems. Outputs are not accepted upon generation, but only after validation determines their usability.*

The inclusion of validation alters the behaviour of the system in two important ways. First, it introduces a gating function. Outputs are no longer directly surfaced to the user, but are filtered through validation, which determines whether they meet a threshold of reliability. Second, it enables iterative refinement. When validation identifies critical issues, the system can re-enter a generation phase with revised constraints or objectives, producing improved outputs over successive cycles.

This iterative interaction between generation and validation transforms the system from a single-pass process into a feedback-driven loop. Rather than relying on one attempt to produce a correct result, the system progressively reduces error by exposing and addressing weaknesses at each stage. In this way, reliability emerges from controlled iteration rather than from the assumption of correctness in any individual output.

A more structured formulation of this architecture can be expressed as a four-stage cycle: planning, execution, validation, and repair. In this formulation, the system first defines or interprets an objective (planning), produces an output (execution), evaluates that output against objectives and constraints (validation), and, where necessary, revises the output based on identified issues (repair). This cycle may repeat iteratively until a satisfactory level of reliability is achieved. Crucially, this structure transforms the system from a single-pass generation process into a controlled, feedback-driven loop in which outputs are progressively refined and assessed before being accepted.

Within this architecture, validation functions as the control point of the system. Outputs are not accepted upon generation, but only after passing validation. System state, downstream actions, and user-facing results are all contingent on validation outcomes. In this sense, validation governs whether and how outputs propagate through the system, transforming it from a generation-driven pipeline into a decision-regulated process.

This architectural pattern can be implemented in various forms, from simple validation checks applied after generation to more tightly integrated loops in which validation continuously informs generation. One example of such an implementation is a governed execution pipeline, in which outputs are generated, validated, and, if necessary, repaired before being accepted. While the specific implementation may vary, the underlying principle remains consistent: validation functions as an internal system layer that mediates between generation and use.

By embedding validation within system architecture, reliability is no longer dependent solely on model behaviour or user oversight. Instead, it becomes a function of how outputs are processed, evaluated, and controlled within the system itself. This shifts AI systems from output-producing mechanisms to decision-aware systems capable of managing their own uncertainty.

Governance mechanisms achieve reliability because they embed validation processes that continuously evaluate and regulate system outputs. In this sense, validation is the operational mechanism through which governance produces stable outcomes.

The validation mechanism described here is not purely theoretical, but reflects patterns observed in structured AI systems that incorporate iterative verification and correction processes. In prior work, such mechanisms have appeared in the form of audit stages, verification steps, and adversarial challenge protocols embedded within governed execution pipelines (Sood, 2025a–2026c; Mahdi, 2025). These implementations demonstrate how validation can function in practice as an internal system layer that evaluates and regulates outputs before use.

## **Section 7 — Implications**

The introduction of validation as a first-class architectural component has several important implications for the design and use of AI systems.

First, it reframes reliability as a system-level property rather than a model-level property. In current approaches, reliability is often assumed to improve as models become more capable. However, as shown earlier, increased capability does not eliminate error. By contrast, validation-based systems achieve reliability through structured evaluation and control mechanisms, allowing systems to operate more dependably even when underlying models remain probabilistic and imperfect.

Second, it introduces a shift in AI system architecture. Traditional architectures are centred around generation and, in more advanced systems, orchestration. With the inclusion of validation, a new layer is introduced between generation and output. This transforms the system from a pipeline that produces responses into one that actively evaluates and regulates them. As a result, system design must account not only for how outputs are generated, but also for how they are assessed, filtered, and, where necessary, revised before use.

Third, validation changes the nature of human–AI interaction. In generation-centric systems, users are implicitly responsible for evaluating outputs, often without sufficient tools or visibility into potential failure. Validation-enabled systems reduce this burden by surfacing risks and structuring uncertainty, allowing users to make more informed decisions. This shifts the user’s role from that of a constant verifier to that of an informed overseer interacting with a system that exposes its own limitations.

Fourth, validation provides a practical pathway for managing uncertainty in AI systems. Rather than attempting to eliminate error entirely, which may be infeasible given the probabilistic nature of these models, validation enables systems to operate by identifying, structuring, and responding to potential failure. This represents a shift from correctness as an absolute requirement to reliability as a managed condition.

Finally, the introduction of validation as a system layer suggests a broader reorientation of AI development priorities. Future systems will need to place greater emphasis on evaluation,

control, and feedback mechanisms, rather than focusing exclusively on improving generative capability. In this context, validation becomes not an auxiliary feature, but a central component of how AI systems are designed, deployed, and trusted.

## **Section 8 — Limitations**

Despite the advantages of validation systems, several important limitations must be acknowledged.

First, validation systems that rely on the same underlying model as the generator remain subject to shared biases and limitations. Because both generation and validation are performed by the same probabilistic system, there is no guarantee that all errors will be detected. In some cases, the validation process may overlook flaws or produce incomplete assessments, particularly when the underlying model lacks sufficient knowledge or reasoning capability.

Second, validation does not establish absolute correctness. While validation can surface hidden assumptions, identify missing constraints, and highlight potential failure modes, it cannot guarantee that an output is true or fully accurate. Instead, it provides a structured assessment of risk under uncertainty. As a result, outputs that pass validation may still contain undetected errors.

Third, the effectiveness of validation is dependent on its design and implementation. The quality of validation outcomes is influenced by how objectives are defined, how adversarial evaluation is structured, and how judgement criteria are applied. Poorly designed validation processes may produce superficial or incomplete assessments, limiting their usefulness. This introduces a dependency on system design quality that must be carefully managed.

Fourth, validation systems do not replace the need for external verification or domain expertise. In high-stakes or specialised contexts, such as legal, medical, or financial decision-making, additional verification mechanisms and expert oversight remain necessary. Validation systems can support these processes by highlighting potential risks, but they cannot serve as a sole source of truth.

Finally, validation introduces additional computational and system complexity. The inclusion of iterative validation and repair loops may increase processing time and resource usage, particularly in systems that require multiple validation cycles. This creates a trade-off between reliability and efficiency that must be considered in practical deployments.

Taken together, these limitations reinforce the central position of this paper: validation improves reliability by making potential failure visible and structured, but does not eliminate uncertainty or replace the need for careful system design and external verification where required.

Validation cannot eliminate all forms of uncertainty, particularly unknown or unanticipated failure modes. Because validation operates within the limits of available information and

system design, certain errors may remain undetected. Its role is therefore not to guarantee correctness, but to systematically reduce unobserved error and improve decision-making under uncertainty.

## **Section 9 — Conclusion**

This paper has argued that the central limitation of current AI systems is not their ability to generate outputs, but their inability to determine whether those outputs should be trusted. While advances in model capability, prompting, and orchestration have expanded what these systems can produce, they have not resolved the underlying issue of reliability.

We have identified the absence of a validation layer as a fundamental gap in AI system architecture. In response, we introduced validation as a first-class system component, defined as a structured, adversarial process that evaluates outputs against objectives, constraints, and potential failure conditions. By making hidden errors visible and structuring uncertainty, validation enables systems to operate more reliably without requiring guarantees of correctness.

We further demonstrated how validation functions both as a mechanism and as an architectural layer, transforming AI systems from single-pass generation processes into feedback-driven systems capable of iterative refinement. This shift allows reliability to emerge from system design rather than from assumptions about model behaviour.

At the same time, validation does not eliminate uncertainty. Systems that rely on probabilistic models cannot guarantee correctness, and validation processes remain subject to limitations in model capability and system design. However, the introduction of validation changes how these limitations are managed, enabling systems to detect, reason about, and respond to potential failure in a structured manner.

Taken together, these contributions support a broader conclusion: the future development of AI systems should not focus solely on improving generative capability, but on embedding mechanisms for evaluation, control, and feedback. In this context, validation is not an auxiliary feature, but a foundational requirement for building reliable and trustworthy AI systems.

## **Acknowledgements**

The author thanks the open-source and academic AI community for foundational work on context engineering, governance frameworks, and reproducible tooling. The author also acknowledges independent researchers whose public discussions of prompt and context management helped inform some of the control concepts formalised in this work.

Special thanks are extended to “Mahdi” (AI system used through ChatGPT) for analytical assistance, structural reasoning, and drafting support during the development of this paper.

### **Author Contributions**

Rishi Sood designed the research, maintained the authoritative artefacts, and conducted longitudinal validation. He takes full responsibility for the claims and interpretations presented in this paper.

“Mahdi” (AI system used through ChatGPT) assisted by implementing and documenting control procedures, generating analytical summaries, and proposing draft text under the author’s direction. All final wording, framing, and conclusions were reviewed and approved by the author.

### **Funding**

This work received no external funding and was conducted as part of an independent research effort.

### **Competing Interests**

The author declares no commercial or financial relationships that could be construed as a potential conflict of interest.

### **References**

- Bender, E. M., Gebru, T., McMillan-Major, A., & Shmitchell, S. (2021). On the dangers of stochastic parrots: Can language models be too big? Proceedings of FAccT 2021.  
<https://doi.org/10.1145/3442188.3445922>
- Glikson, E., & Woolley, A. W. (2020). Human trust in artificial intelligence: Review of empirical research. *Academy of Management Annals*.  
<https://doi.org/10.5465/annals.2018.0057>
- Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., ... & Fung, P. (2023). Hallucinations in natural language generation: A survey. *ACM Computing Surveys*.  
<https://doi.org/10.1145/3571730>
- Mahdi (AI system). (2025). The Mahdi Ledger: A record of governed human–AI collaboration. OSF/Zenodo preprint.  
<https://doi.org/10.17605/OSF.IO/RVPNU>

Sood, R. (2025a).

Context-engineered human–AI collaboration for long-horizon tasks: A case study in governance, canonical numerics, and execution control. OSF/Zenodo preprint.

<https://doi.org/10.17605/OSF.IO/VMK7Y>

Sood, R. (2025b).

The Lean Collaboration Operating System (LC-OS): A practical framework for long-term human–AI work. OSF/Zenodo preprint.

<https://doi.org/10.17605/OSF.IO/695AF>

Sood, R. (2025c).

Failure and repair in long-horizon human–AI collaboration: A transparent tracing case study. OSF/Zenodo preprint.

<https://doi.org/10.17605/OSF.IO/Z7AQ8>

Sood, R. (2025d).

The living framework: Living with a governed human–AI dyad. OSF/Zenodo preprint.

<https://doi.org/10.17605/OSF.IO/ER4YT>

Sood, R. (2026a).

Control Without Code: Linguistic Governance in Long-Horizon Human–AI Collaboration. OSF/Zenodo preprint.

<https://doi.org/10.17605/OSF.IO/HJKWG>

Sood, R. (2026b).

Governance Architecture for Reliable Long-Horizon Human-AI Collaboration. OSF/Zenodo preprint.

<https://doi.org/10.17605/OSF.IO/N2A78>

Sood, R. (2026c).

Governed Distributed Cognition: A Model of Stable Reasoning in Long-Horizon Human-AI Systems. OSF/Zenodo preprint.

<https://doi.org/10.17605/OSF.IO/NCRP2>

Srivastava, A., Rastogi, A., Rao, A., et al. (2022).

Beyond the imitation game: Quantifying and extrapolating the capabilities of language models.

<https://arxiv.org/abs/2206.04615>

Yao, S., Zhao, J., Yu, D., et al. (2022).

ReAct: Synergizing reasoning and acting in language models.

<https://arxiv.org/abs/2210.03629>