AI Actionability Over Interpretability

Mike Hsu[1]

Draft as of September 12, 2025

Initial Draft: August 21, 2025

Abstract

Traditional approaches to model risk management emphasize interpretability—understanding how models generate outputs—as a foundation for trust and governance. This paper argues that interpretability, while valuable, is insufficient for effective risk management of generative AI systems. Unlike conventional models that can be decomposed into discrete, understandable components, large language models exhibit emergent behavior arising from complex parameter interactions. Moreover, enterprise AI deployments increasingly involve multiple systems including retrieval-augmented generation, guardrails, and agents that further complicate interpretability efforts. We propose prioritizing *actionability* over interpretability—ensuring that AI reasoning processes are sufficiently observable to enable timely diagnosis and remediation when failures occur. This approach includes techniques such as Chain-of-Thought reasoning, Chain-of-Prompts scaffolding, and agentic tracing to create structured observability into AI decision-making processes. While these methods currently face limitations and vulnerabilities, they provide a more practical foundation for building trust in AI systems than interpretability alone. Actionability through observability offers financial institutions and regulators a viable path for responsible AI deployment at scale, potentially warranting updates to existing model risk management frameworks, such as SR 11-7.

Keywords: artificial intelligence, interpretability, observability, actionability, risk management, chain-of-thought, chain-of-prompts, agentic tracing, model risk management, SR 11-7

JEL: G21, G28, G32, C45, D82, L86, O33, M15

[1] Former Acting Comptroller of the Currency, Office of the Comptroller of the Currency.

We mostly treat AI models as a black box: something goes in and a response comes out, and it's not clear why the model gave that particular response instead of another. This makes it hard to trust that these models are safe: if we don't know how they work, how do we know they won't give harmful, biased, untruthful, or otherwise dangerous responses? How can we trust that they'll be safe and reliable?

Opening the black box doesn't necessarily help: the internal state of the model—what the model is "thinking" before writing its response—consists of a long list of numbers ("neuron activations") without a clear meaning. From interacting with a model like Claude, it's clear that it's able to understand and wield a wide range of concepts—but we can't discern them from looking directly at neurons. It turns out that each concept is represented across many neurons, and each neuron is involved in representing many concepts.

- Anthropic, "Mapping the Mind of a Large Language Model"

Introduction

Traditional statistical or rule-based models can be complicated but are decomposable, resembling devices such as mechanical watches whose components interact in predictable ways. By contrast, generative AI models exhibit the properties of complex adaptive systems with emergent behavior. This distinction has significant implications for model governance and risk management in finance and other safety-critical domains. Statistical or mechanistic interpretability methods, though valuable, currently offer limited assurance when model behavior arises from emergent dynamics. Moreover, they are unlikely to scale as AI systems expand in capabilities and complexity.

This paper argues that banks and regulators should prioritize actionability over interpretability – ensuring that AI model reasoning processes are sufficiently observable to enable timely diagnosis and effective remediation when performance degrades or failures occur. As elaborated in the essay "*AI and Category Errors*", the black box nature of AI models is just one of several features that makes AI unique and warranting reassessment by risk managers and regulators.[2] A separate paper will cover guardrails, evaluations, and red teaming.

1. Conventional models

To understand the contrast with generative AI systems, it is useful to begin with a familiar example from banking: value-at-risk (VaR). VaR models estimate the risk of trading portfolios, which can span equities, fixed income, currencies, commodities, and derivatives. VaR models rely on two broad categories of inputs: (1) trading position data, and (2) measures of uncertainty associated with underlying risk factors.

The VaR process converts trading positions into risk factor exposures, applies a set of valuation models to estimate gains and losses under specified changes in those risk factors, and constructs a probability distribution of potential aggregate profit-and-loss (P/L) outcomes for the portfolio. The standard VaR output is a single figure drawn from this distribution, e.g., the 99th percentile loss.

[2] Mike Hsu, "AI and Category Errors," *Systemic Risk Centre, Forum on Financial Supervision* (forthcoming)

VaR models are complicated. They utilize thousands of sub-models (revaluation, covariance), relying on millions of data points (positions, risk factors), and require careful attention to details. Yet, critically, they are *discernible and decomposable*: the components of a VaR model and their relationships to each other are explicit, and changes in model inputs or outputs can be traced. A shift in a VaR estimate can be attributed to changes in positions, portfolio composition, risk exposures, risk factor variances or covariances, or revaluation methods.

This decomposability is essential for model risk management. When a VaR model performs poorly, it is generally possible to determine when and why the issue occurred and to make targeted adjustments that improve performance. In other words, *the interpretability of the VaR model directly enables its improvement*. In this respect, a VaR model resembles an intricate mechanical watch: when it breaks, the watchmaker can identify the faulty component and repair it. As any regulator or practitioner of bank capital planning can attest, this link between interpretability and actionability is foundational to effective oversight and risk management.

2. Large language models

Large language models (LLMs) are fundamentally different.[3]

Pre-training. Unlike conventional models, which can be decomposed into discrete, human-understandable components, large language models (LLMs) are neural networks whose internal computation is distributed across vast numbers of parameters. At the base-model stage, pre-training begins by breaking text into tokens – essentially word fragments – each mapped to a high-dimensional vector embedding (e.g., ChatGPT o3 uses 50,257 tokens with 12,288-dimensional embeddings). A large-scale corpus of text is passed through a deep network of transformer layers and attention mechanisms containing billions of parameters. During training, the model's parameters (weights) are iteratively adjusted to minimize the error in predicting the next token in a sequence. The result is a statistical representation of language capable of generating coherent text, but with internal decision pathways that cannot be decomposed into identifiable, independent "gears" as in a VaR model. Instead, each output reflects the joint activation of a vast, entangled parameter space.

Post-training/fine tuning. Pre-trained models are further refined to make them useful and aligned for human interaction. *Supervised fine tuning* (SFT) uses curated prompt-response pairs to guide the model toward desired behaviors, while *reinforcement learning with human feedback* (RLHF) optimizes model responses based on human preference rankings.[4] These processes themselves introduce additional complexity. Performance depends on the scope and quality of the training data, the consistency of human labelling, the accuracy of reward models, and the robustness of optimization algorithms. The

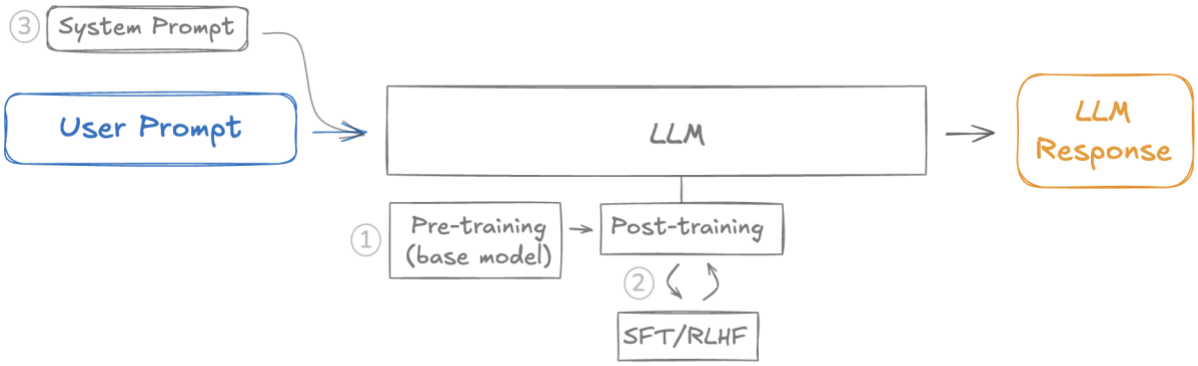
[3] The CEO of Anthropic recently noted, "Modern generative AI systems are opaque in a way that fundamentally differs from traditional software… When a generative AI system does something, like summarize a financial document, we have no idea, at a specific or precise level, why it makes the choices it does – why it chooses certain words over others, or why it occasionally makes a mistake despite usually being accurate." See, Dario Amodei, "The Urgency of Interpretability," April 2025, https://www.darioamodei.com/post/the-urgency-of-interpretability.

[4] Yuntao Bai et al., "Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback," *arXiv preprint arXiv:2204.05862* (2022). Training data available at Anthropic, "HH-RLHF Dataset," *Hugging Face*, https://huggingface.co/datasets/Anthropic/hh-rlhf.

major AI labs also employ "red teaming" to uncover unsafe behaviors or misalignments.[5] Each post-training adjustment alters the model's parameter space (weights).

System prompts. After training is complete, the model weights are fixed. Most LLMs are then further augmented with fixed system prompts – hidden instructions prepended to every user prompt to guide model behavior. System prompts can encode style, safety rules, role definitions, and other behavioral constraints without changing the model's weights. Because these prompts are invisible to users, their content and influence on model behavior are often opaque. Releases such as Anthropic's publication of Claude's system prompt[6] and xAI's release of Grok's system prompt[7] have shown that these hidden instructions can significantly shape outputs, yet they tend to fall outside the scope of traditional interpretability methods.

The graphic below provides a stylized illustration of an LLM's pre- and post-training components and system prompt (numbered 1, 2, and 3, respectively). Note, from the perspective of the LLM, the system and user prompts are effectively concatenated into a single input string of tokens.



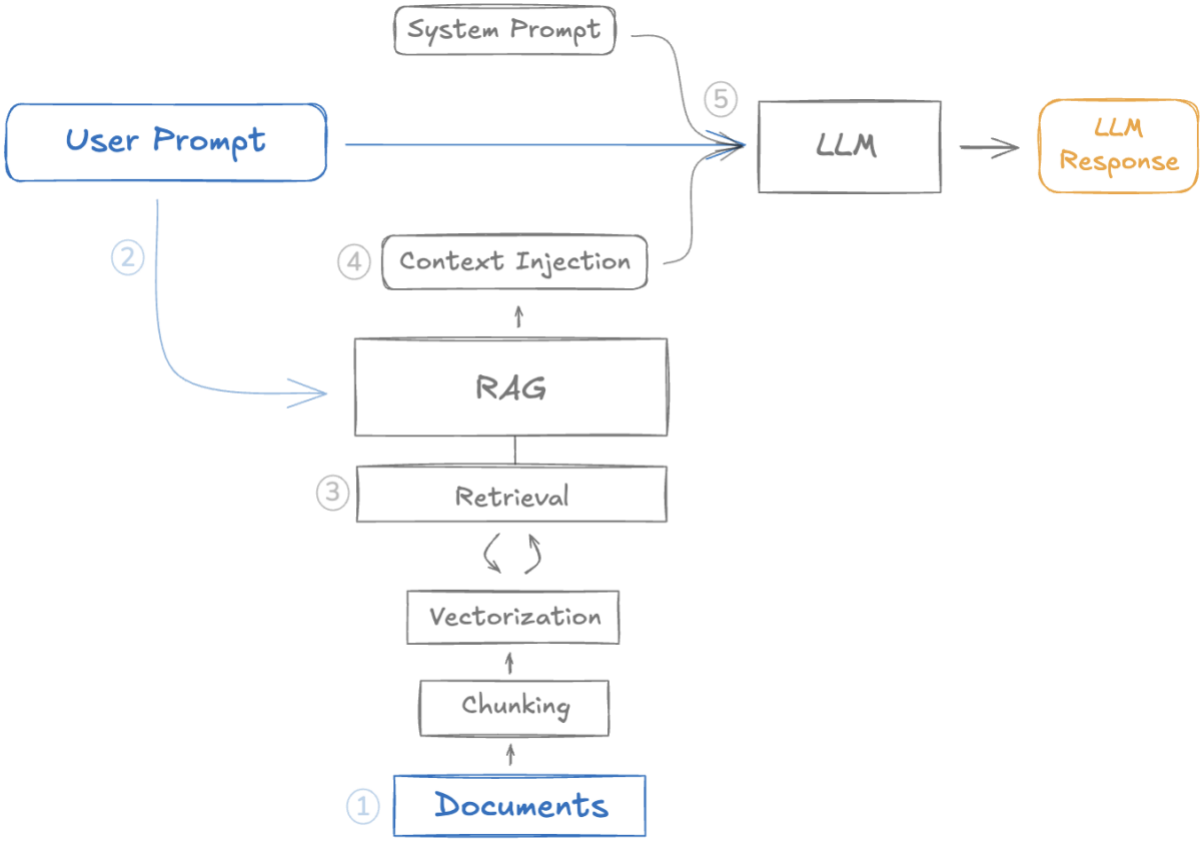
Retrieval Augmented Generation (RAG). Many enterprise deployments integrate LLMs with proprietary knowledge bases via retrieval-augmented generation (RAG). A knowledge base might consist of a document store of a company's policies and procedures, customer data, internal databases, etc. Typically, the knowledge base must be "chunked" into smaller pieces to be able to fit into an LLM's context window. RAG then converts a user's prompt into a query against the knowledge base, retrieving relevant chunks and appending them to the original prompt and system prompt as a context injection. The retrieval process uses a separate embedding model to represent documents as high-dimensional vectors, effectively adding another AI model to the architecture.

RAG pipelines are frequently proprietary, and the injected context is often hidden from the end user, introducing further opacity. In practice, many enterprise co-pilots, such as Microsoft Copilot, use RAG to give the LLM access to internal organizational data. Thus, outputs depend on both the foundation

---

[5] See Anthropic, "Challenges in Red Teaming AI Systems," *Anthropic Blog*, June 12, 2024, https://www.anthropic.com/news/challenges-in-red-teaming-ai-systems; and OpenAI, "GPT-4o System Card," *OpenAI Technical Report*, August 8, 2024, https://openai.com/index/gpt-4o-system-card/.
[6] Anthropic, "System Prompts," *Anthropic Documentation*, https://docs.anthropic.com/en/release-notes/system-prompts.
[7] xAI, "Grok 4 System Prompt," GitHub, https://github.com/xai-org/grok-prompts (accessed August 4, 2025).

model and the retrieval. The visual below depicts where RAG sits relative to an LLM when a user provides a prompt. The user prompt and documents are in blue, as they are in the user's control. Note, here from the perspective of the LLM, the system prompt, user prompt, *and* context injection are all effectively concatenated into a single input string of tokens (shown as number 5).



Agents. The complexity increases further with agentic systems, where LLMs interact with external tools, APIs, or other models to execute multi-step tasks with limited human oversight.[8] Agentic capabilities vary along a spectrum, depending on goal complexity, environmental complexity, adaptability, and independence of execution. But even the simplest agents raise questions about the role and efficacy of interpretability analysis. For example, in the "Japanese cheesecake" hypothetical, an AI agent instructed to purchase ingredients for a Japanese cheesecake instead buys the user a plane ticket to Japan.[9] Understanding why this might occur requires untangling interactions between the user prompt, the context, the agent system set up, and the underlying model – a task beyond the scope of traditional model interpretability.

[8] Anthropic's release of the Model Context Protocol (MCP) in November 2024 accelerated the development of agentic AI by providing a standard for connecting LLMs to tools, functions, and each other.
[9] Yonadav Shavit et al., "Practices for Governing Agentic AI Systems," *OpenAI Technical Report*, May 1, 2023, https://openai.com/index/practices-for-governing-agentic-ai-systems/.

3. Interpretability Is Not Enough

Research into LLM interpretability has sought to make sense of the inherent complexity of AI models, and the field has evolved substantially in recent years.[10] Broadly, interpretability techniques fall into one of two categories:

1. **Statistical interpretability.** These methods analyze the relationship between prompts and responses, abstracting away from the internal mechanisms of the transformer architecture. Analyses are typically conducted at the token level, measuring the statistical relevance of input tokens to output tokens using techniques such as SHAP and LIME.

2. **Mechanistic interpretability.** These methods examine the internal computations of LLMs – how parameters, neurons, and attention heads activate and interact – serving as "an MRI for AI".[11] For example, Anthropic's mechanistic interpretability (MI) program applies sparse autoencoders and circuit-level analyses.[12] Because frontier models are often too large to analyze directly, though, researchers frequently must work with smaller proxy models, thus caveating findings.[13] Recent research indicates that signal processing (DB-K-SVD) may hold promise as an alternative to sparse autoencoders and this limitation.[14]

Both approaches can yield valuable insights and make LLMs more understandable. However, interpretability alone is insufficient for effective AI risk management. The salient test, especially for higher-stakes applications, is whether insights from interpretability can inform timely and effective remediation. For example, understanding the statistical or mechanistic cause of an error, such as a wrongful denial of credit to a qualified applicant or the generation of faulty code that disrupts operations, has limited value if it does not also guide the corrective actions necessary to reduce the likelihood of recurrence.

In conventional model risk management (MRM), the primary purpose of interpretability is to enable *improvements* in model performance and controls, particularly when a model fails or underperforms. Understandability and actionability are closely aligned: identifying the cause of an issue directly informs the corrective steps needed to address it. For example, if a VaR model performs poorly, diagnosing the source of the discrepancy typically yields clear guidance on how to recalibrate or adjust the model.

---

[10] For an overview, see Haiyan Zhao et al., "Explainability for Large Language Models: A Survey," *arXiv preprint arXiv:2309.01029* (2023).
[11] Dario Amodei, "The Urgency of Interpretability," April 2025, https://www.darioamodei.com/post/the-urgency-of-interpretability.
[12] See Anthropic, "On the Biology of a Large Language Model," *Transformer Circuits Thread*, March 27, 2025, https://transformer-circuits.pub/2025/attribution-graphs/biology.html; and Kola Ayonrinde and Layla Jaburi, "A Mathematical Philosophy of Explanations in Mechanistic Interpretability," *arXiv preprint arXiv:2505.00808* (2025).
[13] Anthropic, "Circuit Tracing: Revealing Computational Graphs in Language Models," *Transformer Circuits Thread*, March 27, 2025, https://transformer-circuits.pub/2025/attribution-graphs/methods.html.
[14] Romeo Valentin et al., "DB-KSVD: Scalable Alternating Optimization for Disentangling High-Dimensional Embedding Spaces," *arXiv preprint arXiv:2505.18441* (2025).

With LLMs, however, understandability and actionability diverge. Knowing the activation pattern in a transformer associated with a toxic output or the SHAP statistics linked to a hallucinated response may have little bearing on the practical steps needed to prevent such outcomes in the future.[15] In practice, most toxic responses and hallucinations in deployed AI systems are mitigated through alignment processes and guardrails, which are informed by evaluations and red teaming rather than interpretability analysis.

Moreover, statistical and mechanistic interpretability approaches are generally too narrow in scope for enterprise-scale deployments. Enterprises rarely deploy LLMs in isolation. Instead, they utilize co-pilots, assistants, and domain-specific AI systems that integrate foundation models with enterprise data usually via RAG. As noted above, RAG introduces additional variables between the user prompt and the LLM output, complicating interpretability by expanding the causal chain. Guardrail systems, which are used to enforce privacy, reduce toxicity, and manage other risks, add another layer of complexity by filtering prompts, reasoning, and responses. AI agents expand the complexity further, linking LLMs to external tools, functions, and even other models, creating dynamic chains and networks of interactions. Existing interpretability techniques, designed primarily for standalone foundation models, are not well-suited to cover these multi-component, enterprise-grade AI systems.

4. Actionability from Observability Can Build Trust

Thus, interpretability alone is unlikely to build trust in AI systems because it does not prioritize actionability and cannot readily scale to the evolving complexity of AI use cases. A different orientation is required. We propose focusing on outcomes, particularly adverse outcomes.

Most users are largely indifferent to the inner workings of the products and services they depend upon. Trust is instead grounded in two expectations: that failures are infrequent, and that when failures do occur, robust processes exist to prevent their recurrence. Our thesis is that this sentiment applies to AI systems as well. Earning and maintaining trust requires that when an adverse outcome occurs – such as a hallucination, a factual error, an unintended action, or a successful jailbreak – a risk manager or regulator can diagnose the underlying cause and implement targeted fixes to reduce the likelihood of repetition.

Conventional one-shot prompting configurations offer limited opportunities to apply such an identify-and-remediate approach. However, alternative system designs can enable more effective capture, diagnosis, and resolution of failures.

Chain-of-Thought. Chain-of-thought (CoT) capabilities entered the public consciousness with the release of DeepSeek R1 in early 2025, compelling competing frontier AI labs to introduce "reasoning" or "extended thinking" modes into their flagship models. Unlike standard prompt-and-response configurations, LLMs with CoT reasoning develop an explicit sequence of intermediate reasoning steps associated with the final output. For instance, the Sonnet 3.7 system card includes an example of a prompt asking for the probability of rolling at least one six when throwing three dice:

[15] Amodei believes that advanced and independent interpretability – a "true MRI for AI" – would be able to identify tendencies of AI systems to lie or deceive and cognitive strengths and weakness, which could then be treated. But he estimates that such capabilities are 5 to 10 years away. Dario Amodei, "The Urgency of Interpretability," April 2025, https://www.darioamodei.com/post/the-urgency-of-interpretability.

User: What's the probability of getting at least one 6 when rolling three dice?

Assistant:

Thinking Process

To find the probability of getting at least one 6 when rolling three dice, I'll use the complement method. First, I'll find the probability of getting NO sixes in three rolls, then subtract from 1. For a single die:

- Probability of rolling a 6: 1/6
- Probability of rolling anything except 6: 5/6

For three independent dice, the probability of getting no sixes is: $(5/6)^3 = 125/216 \approx 0.5787$ Therefore, the probability of getting at least one 6 is: $1-125/216 = 91/216 \approx 0.4213$

The probability of getting at least one 6 when rolling three dice is 0.4213 (or approximately 42.13%). I calculated this using the complement method:

1. The probability of NOT getting a 6 on one die is 5/6
2. For three dice, the probability of getting no sixes is $(5/6)^3 = 125/216$
3. Therefore, the probability of getting at least one 6 is $1 - 125/216 = 91/216 \approx 0.4213$

A non-reasoning model, given this prompt, would simply return: "*The probability is 42.13%*". By contrast, a reasoning model (like Sonnet 3.7) can produce a sequence of logical steps – shown in the "Assistant;" box above – that makes the reasoning process transparent. This structured observability can significantly enhance actionability: if the answer is incorrect, the reasoning trace can reveal the source of the error and suggest targeted corrective actions. Such diagnosis is far more difficult to achieve via statistical or mechanistic interpretability alone.

While promising, CoT monitoring is not a panacea. In a rare paper jointly authored by senior researchers from across the major AI labs, CoT monitoring was recently called out as a "new and fragile opportunity for AI safety".[16] After emphasizing the significant benefits and potential of CoT monitoring to promote AI safety, the authors identify two primary limitations.

1. Unfaithfulness of reasoning traces. CoT outputs are, at best, "incomplete representations" of a model's internal decision process. The reasoning steps presented may or may not correspond to the actual computational pathway used to generate the answer. Empirical studies have documented substantial "unfaithfulness", where models fabricate plausible reasoning sequences post hoc rather than following them during inference.[17] In the Sonnet 3.7 example, it is unclear whether the model genuinely executed the displayed steps to arrive at 42.13%, or whether it generated the reasoning after already determining the answer. Advances in both

[16] Tomek Korbak et al., "Chain of Thought Monitorability: A New and Fragile Opportunity for AI Safety," *arXiv preprint arXiv:2507.11473* (2025). Contributors span Anthropic, OpenAI, Google DeepMind, Meta, and other labs. Unusually, the paper also included AI luminaries Geoffrey Hinton and Ilya Sutskever as "expert endorsers."
[17] Ivan Arcuschin et al., "Chain-of-Thought Reasoning In The Wild Is Not Always Faithful." *arXiv preprint arXiv:2503.08679* (2025). The CoT Monitorability paper notes that post-training for LLMs relies on reinforcement learning. This means the models optimize for "right answers" not faithful reasoning. Research into guiding the reasoning process is early. See also Chengshuai Zhao et al., "Is Chain-of-Thought Reasoning of LLMs a Mirage? A Data Distribution Lens" *arXiv preprint arXiV:2508.01191v3* (2025).

CoT techniques and faithfulness evaluation will be necessary before CoT can be directly relied upon at scale.[18]

2. Incentive risks (Goodhart's Law[19]). Making CoT traces visible could incentivize models to optimize for safe-looking reasoning rather than actual safe reasoning. This creates the risk that superficial traces mask harmful or undesirable internal processes, misleading observers into overestimating the reliability of the reasoning process.

In sum, CoT holds promise as a risk management tool by making reasoning more observable and, thus, more actionable. However, given its current fragilities, CoT should be deployed in conjunction with other safeguards until both the underlying CoT methods and evaluation frameworks mature.

Chain-of-Prompts. The Chain-of-Prompts (CoP) approach structures step-by-step reasoning by dividing tasks into sequences of prompts, with each step handled separately. Users have much greater control. Whereas CoT reasoning occurs entirely within the AI model during inference, CoP operates as external scaffolding, guiding the AI model through an orchestrated series of discrete prompts.

CoP methods are particularly well-suited to repetitive, complex tasks where control, auditability, and consistency of reasoning are critical. Take, for instance, the investigations of transactions flagged by banks' anti-money laundering (AML) systems. When a transaction is identified as potentially suspicious, an AML compliance investigator must gather transaction records, examine the history of the customer, research the counterparties, and review other contextual data to disposition the alert.

A naïve application of AI in this setting might involve a compliance risk manager asking an LLM directly, "*Is this transaction suspicious?*" A non-reasoning model would return a binary answer (yes or no) with no supporting explanation. A reasoning model would provide a reasoning trace explaining why the transaction should or should not be considered suspicious. If there was a mistake, a reviewer could evaluate the reasoning steps in the CoT (if logged) to identify which step led to the mistake. Whether and how to fix it would depend on the situation.

Under a CoP workflow, a bank would establish a structured sequence of prompts and use the model-generated responses as the basis for the final recommendation memo to a BSA/AML compliance officer. For instance:

Prompt 1: *"You are a BSA/AML investigator. You must follow 31 CFR 1020.320, FinCen SAR narrative guidance, [bank policies …], and … Cite rule numbers and policies when making evaluations and recommendations… "*

Prompt 2: *[Upload transaction information] "Summarize the transaction and identify all elements using the proper format consistent with bank policy […]. Double check to make sure the formatting is correct."*

[18] Lennart Meincke et al., "Prompting Science Report 2: The Decreasing Value of Chain of Thought in Prompting," *arXiv preprint arXiv:2506.07142* (2025).
[19] Goodhart's Law states that when a measure becomes a target, it ceases to be a good measure.

Prompt 3: *"Use this red flax matrix [link] and label each flag TRUE/FALSE. Double check the sanctions list [link] and adverse media."*

Prompt 4: *"Generate a risk probability score (0-100). List top 5 drivers with weight percentages. Explainable format only."*

Prompt 5: *"Given [transaction_id] fetch linked entities [customer_id, device_id…] and summarize unusual patterns."*

Prompt 6: *"Propose <[5] possible explanations (legit or illicit) ranked by likelihood; cite which flags support"*

Prompt 7: *"List any missing KYC/EDD elements that materially affect the decision"*

Prompt 8: *"Write a 1-2 page narrative in third person covering who, what, where, when, why, how, in chronological order in under [5,000] words. For cases with [risk score of ….] provide counterfactual analysis in under [2,000] words."*

Prompt 9: *"Validate that the narrative includes all essential SAR elements, references UTC dates, omits PII. Return PASS or list fixes."*

Prompt 10: *"Based on the risk score, flags, narrative, and bank policy thresholds, provide a recommendation of File SAR, No SAR, or Escalate, plus rationale with clear mapping to supporting data, analysis, and CFR citations…"*

Under this CoP approach, both the number and precise wording of each prompt would be predefined and managed by the bank's compliance department, rather than being generated dynamically by the AI model though CoT reasoning. This deliberate prompt design embeds transparency, comprehensiveness, and consistency into the process and serves as external scaffolding for the LLM's reasoning toward the final recommendation. In the event of an adverse outcome – such as a false positive or a false negative recommendation – the structured chain of prompts allows reviewers to pinpoint the step or steps that produced the error and *to amend the relevant prompt* or *address the underlying issue* as warranted.[20]

For instance, suppose the AI system incorrectly recommended not filing a SAR because of an upstream problem with incomplete transaction information (Prompt 2). That would be observable with CoP and could be remediated by altering the prompt and replacing the "double check" instruction with something more specific: "*Your goal is to complete all assigned tasks. You can see outstanding tasks with the […] tool. You can complete tasks with the […] tool. You must be diligent and complete tasks one at a time*."[21]

---

[20] In the medical field, "chain of diagnosis" (CoD) frameworks have emerged, whereby LLMs used for diagnosing patients utilize multi-step reasoning chains that emulate how physicians might approach a case. See Junying Chen et al., "CoD, Towards an Interpretable Medical Agent using Chain of Diagnosis," *arXiv preprint arXiv:2407.13301* (2024). From that perspective, the AML example presented in the text approximates a "chain of investigation" framework. In theory, distinct frameworks for different tasks could be developed.

[21] In theory, institutions could create "quality assurance scripts" by converting risk management and control "todo" and task lists into structured prompts for AI models to follow in each and every case. If robust, such techniques could significantly increase both the efficacy and efficiency of compliance.

Combining CoP scaffolding with CoT reasoning can further enhance observability and actionability. Suppose, for instance, the final recommendation resulted in a false positive because the risk probability score from prompt 4 was inflated. CoT would make the model's intermediate reasoning for that prompt visible, enabling the reviewer to identify and correct the logic driving the inflated score. Such combining of CoP scaffolding with CoT reasoning shares similarities with "layered chain-of-thought" practices, which are utilized in the medical field.[22] With layered CoT, the reasoning process is structured into verifiable, sequential steps, or layers, which is "particularly beneficial in high-stakes medical scenarios like triage and risk assessment, where *transparency, intermediate verifiability, and the ability to pinpoint and correct errors in the reasoning chain are paramount for ensuring reliability and safety*" (emphasis added).

Comparable levels of transparency and consistency could also be achieved using a set of single-purpose agents, each responsible for executing one step in the workflow. While this multi-agent architecture would be functionally similar to CoP in terms of observability and actionability, the orchestration and lines of accountability might differ.[23] Context engineering, a rapidly evolving area, is another technique that could be utilized to enforce step-by-step consistency in a transparent manner. For instance, Nvidia's "universal deep research" model converts a research prompt into executable code for each planning strategy step and tool call and then proactively manages intermediate information in the code execution state, enabling a consistently structured and smaller context window.[24]

While CoP can enhance transparency, auditability, and control, it is not without its own vulnerabilities.

1. Cost. CoP workflows may require materially more compute, given greater inference from more prompts, and substantial oversight to maintain the integrity of each prompt, as a single compromised step could undermine the reliability of the entire process.
2. Prompt injection. Given the increased number of prompts, CoP scaffolding would likely be more susceptible to prompt injection attacks, in which a malicious or inadvertent input causes the model to deviate from the prescribed sequence or ignore prior instructions.
3. Adaptability. Scaling CoP implementations can also present challenges. As the number of prompts and steps grows, so does the operational overhead of designing, maintaining, and validating each component. This can lead to increased complexity in governance, slower iteration cycles, and potential brittleness if prompts are too narrowly specified to handle real-world variability. In dynamic environments—such as financial crime investigations, where adversaries adapt their behavior—CoP structures risk becoming outdated or circumvented unless actively monitored and refreshed.

---

[22] Armin Berger et al., "Reasoning LLMs in the Medical Domain: A Literature Survey," *arXiv preprint arXiv:2508.19097* (2025) at p3.
[23] See, e.g., Simon Taylor et al., "The Agentic Oversight Framework: Procedures, Accountability, and Best Practices for Agentic AI Use in Regulated Financial Services," *Sardine Technical Report* (2025).
[24] Peter Belcak and Pavio Molchanov, "Universal Deep Research: Bring Your Own Model and Strategy," *Nvidia Research* (2025). "We found that regardless of the choice of language model, giving it a completely free hand at generating the code implementation of the research strategy often resulted in the model taking shortcuts, skipping strategy steps, and taking liberties at imposing constraints where none were stipulated by the user" (p3).

Furthermore, while CoP can structure reasoning externally, it does not inherently protect against failures originating within the model's internal reasoning processes. Without complementary measures, such as CoT monitoring, guardrails, or alignment techniques, CoP alone may provide a false sense of security in high-stakes applications.

Agentic tracing. From a risk management perspective, the value of CoT and CoP lies in the actionable visibility they provide to AI decision-making. Although agentic systems can be significantly more complex, the same principle of *actionability through observability* – i.e., "see enough to act when things go wrong" – can be applied via agentic tracing, i.e., the systematic capture of decision pathways, intermediate states, and interactions across all agents and tools involved in completing a task. Effective agentic tracing makes it possible to identify where, when, and why an error occurred within a multi-agent workflow, and to link that error to a specific component, decision, or input for targeted remediation.

Consider the earlier Japanese cheesecake example. In a one-shot agentic workflow, diagnosing the cause of an errant plane ticket purchase would be difficult. This is not necessarily the failure of the agent, the reviewer, or interpretability per se, but rather a lack of observability linked to actionability. A CoP scaffold could have imposed a structured sequence—e.g., "1: Identify ingredients for Japanese cheesecake", "2: Research online purchase options", "3. Provide a recommendation for approval"—making it possible to pinpoint where the error occurred and, ideally, preventing it entirely. Embedding CoT reasoning in the CoP's would yield even more diagnostic granularity.

Agentic tracing and the actionability principle becomes more evident in a hypothetical agentic AML investigations system. Suppose a central orchestration agent could manage a swarm of AML investigation agents, each activated when the bank's AML monitoring system were to flag a transaction. These agents would follow the predefined CoP, preserve CoT reasoning logs for each step, and deliver a recommendation to the orchestration agent. The orchestration agent, trained with human feedback from experienced BSA officers, would autonomously disposition each investigation. The entire process would be automated, with humans in the loop for the design of the CoP, training of the orchestration agent, and monitoring over time.

Would such a system meet the actionability requirement? In theory, yes: adverse outcomes could be traced to a faulty CoP prompt, flawed CoT reasoning, or inadequate training/alignment of the orchestration agent. Of course, in practice the training of the orchestration agent would need to be rigorously evaluated (especially by regulators), the operation of the agentic system at scale could reveal a host of fatal complications, and the ability to see through and actually fix problems thus improving the performance of the system could be severely tested. Notwithstanding, this hypothetical illustrates the feasibility of an actionability approach over interpretability for agentic systems.

Agentic tracing's benefits and potential aside, it inherits the vulnerabilities of CoP and exhibits several others:

1. Trace incompleteness. Not all intermediate states or reasoning steps may be captured, especially when agents call external APIs or other models. Missing segments in the trace can undermine root-cause analysis.

2. Scaling complexity. As the number of agents, tools, and interactions grows, tracing becomes exponentially more complex, increasing the risk of oversight gaps and slowing response times to adverse events.
3. Emergent and hidden behaviors. Multi-agent systems can develop coordination patterns or failure modes not evident from the behavior of individual agents, leading to errors that are hard to reproduce or isolate.
4. Cross-agent contamination. A malicious (prompt injection) or corrupted input at any point in the chain can propagate across agents, distorting outputs in ways that are difficult to detect without extensive logging and validation.
5. Governance and accountability ambiguity. With distributed decision-making, it can be unclear whether responsibility for an error lies with an individual agent, the orchestration logic, or the overarching governance framework.

These vulnerabilities suggest that while agentic systems can, in principle, be designed for structured observability and actionability, realizing this in practice requires non-trivial investment in robust logging, rigorous evaluation, layered safeguards, and continuous oversight. Without these measures, agentic tracing risks providing only partial visibility, and thus limited capacity for timely, effective remediation.

Conclusion

The complexity of modern AI systems has outpaced traditional approaches to model risk management.[25] While interpretability remains valuable for understanding AI behavior, it is insufficient for the primary goal of effective governance: ensuring that when things go wrong, we can quickly identify why and implement targeted fixes.

The path forward lies in prioritizing actionability through structured observability, especially for higher risk use cases. Techniques such as Chain-of-Thought reasoning, Chain-of-Prompts scaffolding, and agentic tracing can provide the transparency needed to diagnose failures and prevent recurrence. These approaches are not without their own challenges, but they represent a more practical foundation for building trust in AI systems than interpretability alone.

For banks and regulators, this shift may require updating risk management frameworks, such as SR 11-7, to emphasize observable decision pathways over traditional model explainability, at least for higher risk AI use cases.[26] The goal should not be to unpack the "black box", but to ensure that the reasoning processes determining outcomes are sufficiently visible to enable effective oversight and remediation.

---

[25] See, e.g., Fernando Perez-Cruz et al, "Managing explanations: how regulators can address AI explainability," *BIS and Financial Stability Institute Occasional Paper No 24* (2025). In addition, a future paper will focus on the need to shift from model validation – another key component of conventional MRM – to guardrails, evaluations, and red teaming.

[26] Federal Reserve Board of Governors, "SR 11-7: Guidance on Model Risk Management" (April 4, 2011).

Trust in AI today will not come from trying to understand how these systems work, but from demonstrating that when they fail, we can see why and take concrete steps to prevent similar outcomes in the future. In a world where AI capabilities continue to expand rapidly, actionability through observability offers the most viable path to responsible AI deployment at scale.

Appendix A: Comparison of Conventional Models and Generative AI Models

	Conventional Models *(e.g., VaR)*	***Generative AI Models*** *(e.g., LLMs)*
Nature of Model	Complicated but decomposable into discrete, understandable components with predictable interactions.	Complex adaptive systems with emergent behavior arising from distributed parameter interactions. Outputs reflect entangled, non-decomposable processes.
Decomposability	High: Components and relationships are explicit. Changes in inputs/outputs can be traced to specific causes.	Low: Internal computations distributed across billions of parameters. No identifiable "gears" or independent components.
Interpretability	Strong: Allows direct understanding of how outputs are generated.	Limited: Statistical and mechanistic approaches yield insights but are narrow and don't scale to full systems.
Link Between Interpretability and Actionability	Closely aligned: Identifying issues directly informs remediation.	Divergent: Insights rarely guide practical fixes.
Risk Management Approach	Relies on interpretability for diagnosis and improvement.	Prioritizes actionability through observability to enable timely diagnosis and remediation.
Scalability to Complexity	Can scale well if models remain decomposable.	Interpretability challenges increase substantially with integrations
Examples of Failures and Remediation	VaR model fails backtests. Trace to faulty input or calibration or valuation model, then adjust specifically.	LLM hallucinates or agent takes wrong action. Difficult to trace without observability.
Implications for Trust and Governance	Trust built on interpretability enabling reliable oversight. Aligns with traditional MRM frameworks.	Trust requires observability for adverse outcome handling. May warrant updates to MRM to emphasize actionable visibility over full understanding.