System-Level Observability for Agentic AI Systems: From Model Evaluation to System Assurance

Deepinder Sidhu
Department of Computer Science and Electrical Engineering
University of Maryland, Baltimore County
sidhu@umbc.edu

Abstract

Agentic AI systems are increasingly deployed as operational systems that execute multi-step workflows through interactions among models, tools, services, and external dependencies. However, current evaluation practices remain largely model-centric, focusing on isolated prompt-response interactions under controlled conditions. This creates a fundamental mismatch between evaluation and deployment, as system behavior in operation is governed by execution dynamics, interactions, and runtime variability.

Empirical evidence from real-world deployments shows that failures in agentic AI systems occur during execution and manifest at the system level. These failures arise from orchestration effects, dependency coupling, concurrency, and runtime variability, rather than from incorrect model outputs alone. A key observation is that certain classes of errors remain hidden under model-level evaluation and become observable only when systems are exercised through multi-step execution sequences.

This paper presents a system-level observability framework for evaluating agentic AI systems under realistic operating conditions. The approach integrates structured prompt specifications, test generation, execution of multi-step workflows, and multi-layer observability to analyze system behavior as it occurs in operation. By transforming prompt specifications into execution sequences and evaluating them under controlled yet realistic conditions, the framework enables systematic exposure of both interaction-induced failures and latent model-level weaknesses.

System behavior is captured through packet-level traces (PCAP), system logs, and performance metrics, from which execution-level measurements such as latency, retransmissions, and failure conditions are derived. These provide complementary visibility into execution dynamics and establish an observable basis for analysis. Beyond observation, the framework supports reconstruction of execution pathways to explain how outcomes are produced and enables independent validation through controlled re-execution.

The results establish that model-level correctness does not guarantee system-level reliability and highlight the need for evaluation methodologies that operate on execution sequences under realistic conditions. More broadly, this work positions system-level observability, traceability, and validation as a foundational layer in the AI evaluation stack, enabling principled assessment of agentic AI systems in operational environments.

1 Introduction and Motivation

Agentic AI systems are increasingly deployed as operational systems that perform multi-step tasks through interactions among models, tools, services, and external data sources [1, 2]. Unlike standalone models evaluated in isolation, these systems execute workflows that evolve over time, maintain state, and depend on dynamic runtime conditions. As a result, their behavior is determined not only by model inference but by system-level interactions, orchestration, and execution dynamics.

Current evaluation practices, however, remain largely model-centric. Benchmarks and prompt-response testing focus on isolated interactions under controlled conditions, measuring accuracy, coherence, and task

performance at the level of individual model outputs [3, 4]. While such evaluations are effective for assessing model capabilities, they do not capture the behavior of agentic AI systems as they operate in deployment.

This mismatch between evaluation and deployment has direct consequences. Failures observed in real-world systems are rarely attributable to incorrect model outputs alone [5, 6, 7, 8]. Instead, they arise from interactions across components, dependency coupling, concurrency effects, and variability in execution conditions. These failures manifest during operation, often under load or degraded conditions, and may not be reproducible through isolated testing.

A key observation is that certain classes of errors remain hidden under model-level evaluation and become observable only during system-level execution. When model logic is exercised as part of multi-step execution sequences, latent deficiencies related to timing, interaction effects, and dependency behavior can be exposed. This indicates that model-level correctness does not imply system-level reliability.

These considerations highlight a fundamental gap in current evaluation methodologies. Existing approaches do not systematically capture execution behavior, interaction effects, or runtime variability, and therefore do not provide a sufficient basis for assessing the reliability of deployed agentic AI systems, particularly in mission-critical environments.

This paper addresses this gap by introducing a system-level observability framework for evaluating agentic AI systems under realistic operating conditions. The proposed approach integrates structured prompt specifications, test generation, execution of multi-step workflows, and multi-layer observability to analyze system behavior as it occurs in operation [9, 10]. By transforming prompt specifications into execution sequences and evaluating them under controlled yet realistic conditions, the framework enables systematic exploration of both nominal and failure-inducing scenarios.

System behavior is captured through packet-level traces, system logs, and performance metrics, providing complementary visibility into execution dynamics. These observations enable the derivation of reliability characteristics from observable evidence rather than from model-level assumptions.

The contributions of this paper are as follows:

- We provide empirical evidence that failures in deployed AI systems occur at the system level during execution and are not adequately captured by model-level evaluation.
- We identify the role of execution sequences and interaction effects in exposing both system-level failures and latent model-level weaknesses.
- We present a system-level observability framework that integrates structured test generation, realistic execution environments, and multi-layer observability.
- We demonstrate how reliability can be derived from observable system behavior under realistic operating conditions.

The remainder of the paper is organized as follows. Section 2 presents evidence of failures and analyzes their implications for reliability in agentic AI systems. Section 3 introduces the proposed system-level observability framework and describes how reliability can be derived from execution-level evidence.

2 Failures and Reliability Challenges in Agentic AI Systems

2.1 Agentic AI System Failures

Agentic AI systems are deployed as distributed, execution-dependent systems whose behavior is determined by multi-step workflows, interactions among services, and runtime conditions. In operation, these systems exhibit a range of failure modes that extend beyond model-level behavior.

Platform	Event	Symptoms	Failure Domain	System Insight
Anthropic Claude	Outage	Login failures, API errors, elevated service errors	Control plane, dependency interaction, load instability	Multi-service coupling can disrupt the full service
OpenAI ChatGPT	Repeated outages	Unavailability, latency spikes, degraded responses	Capacity limits, orchestration, resource contention	Burst demand exposes scaling limits and queue buildup
Google Gemini/Bard	Disruption	Partial outages, degraded performance	Backend dependency, rollout issues	Tight coupling increases fragility
Microsoft Copilot	Integration failure	Feature degradation across applications	API dependency, orchestration	External dependencies propagate failures
Amazon Bedrock	API degradation	Timeouts, throttling, reduced throughput	Capacity constraints, scaling	Infrastructure dynamics affect AI reliability

Table 1: Representative AI service failures illustrating recurring system-level failure modes.

Table 1 summarizes representative incidents across multiple AI platforms. These failures are observed under real operating conditions and include service disruptions, latency spikes, integration failures, and state inconsistencies.

Table 1 provides empirical evidence that failures in deployed AI systems occur in operation and manifest at the system level. The observed symptoms arise under real execution conditions and are not attributable to isolated model inference.

Figure 1 illustrates how failures emerge during execution as a result of interaction effects, network conditions, and multi-step workflows. Even when individual components behave correctly in isolation, execution sequences β_k can exhibit degraded behavior due to latency, retransmissions, and dependency interactions.

2.2 Analysis of Failures

The failures summarized above exhibit consistent characteristics across platforms. They are predominantly systemic, arising from orchestration, dependency interactions, resource contention, and execution dynamics rather than from incorrect model outputs.

A key observation is that certain classes of errors remain hidden under model-level evaluation. Model-centric testing operates on isolated prompt-response interactions under controlled conditions, which can mask latent deficiencies related to timing, dependency interactions, and multi-step execution.

When the same logic is executed as part of a sequence β_k under realistic conditions, these masked behaviors become observable. Interaction effects, concurrency, and runtime variability expose both new failure modes and underlying model-level weaknesses that are not apparent in isolation.

This indicates that the dominant failure surface of agentic AI systems lies at the level of execution sequences and system interactions. Model-level correctness does not imply system-level reliability.

2.3 Implications for Reliability in Mission-Critical Systems

Agentic AI systems are increasingly deployed in environments where reliability, timeliness, and consistency are essential. In such settings, failures are not mere degradations in service quality but have direct operational

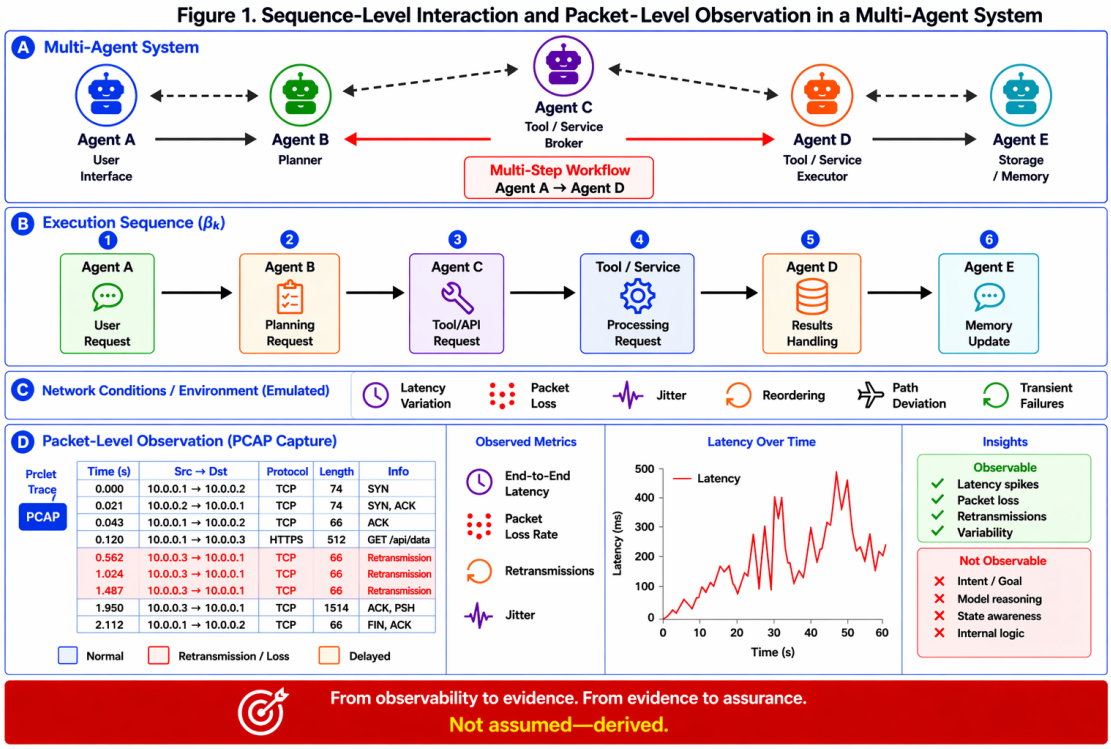

Figure 1: Sequence-level interaction and packet-level observation in a multi-agent system. Execution sequences β_k involve interactions across agents and services under realistic network conditions. Packet-level traces (PCAP) enable the derivation of execution metrics such as latency, retransmissions, packet loss, and jitter, providing visibility into execution dynamics that is not observable under isolated model-level evaluation.

consequences.

The evidence and analysis in the preceding subsections indicate that many failure modes arise from system-level execution, including interaction effects, dependency coupling, and runtime variability. These conditions are largely absent from conventional model-level evaluation pipelines, which operate under isolated and controlled conditions.

As a result, model-level performance metrics do not reliably predict operational behavior. Systems that appear robust under evaluation may exhibit degraded or inconsistent performance in deployment, particularly under concurrency, load variation, and failure scenarios.

In mission-critical domains, such discrepancies are unacceptable. In defense and national security contexts, delays or inconsistencies can disrupt decision processes. In healthcare, system interruptions or inconsistent behavior can affect clinical workflows and access to critical information. In financial systems, latency spikes and partial outages can impact decision quality and operational outcomes.

These observations indicate that reliability must be evaluated at the level of system execution rather than model inference alone. In particular, evaluation must capture execution sequences, component interactions, and behavior under realistic operating conditions.

This establishes a fundamental requirement: reliable deployment of agentic AI systems in mission-critical environments requires system-level evaluation that exposes both interaction-induced failures and latent model-level weaknesses.

3 System-Level Observability for Agentic AI Systems

The preceding section established that failures in deployed agentic AI systems arise during execution and are not adequately captured by model-level evaluation. In particular, reliability cannot be inferred from isolated prompt-response testing, as many failure modes emerge from interactions, dependencies, and runtime conditions.

This section presents a system-level observability framework designed to evaluate agentic AI systems under realistic operating conditions. Observability in AI systems is commonly realized through telemetry such as model inputs and outputs, logs, metrics, and traces, providing visibility into behavior during deployment [9, 10]. These approaches offer valuable insight into system and model performance, but primarily reflect behavior under existing runtime conditions and depend on what is exercised in practice. In contrast, the framework proposed in this work enables observability to be driven by controlled execution, where system behavior is systematically generated through structured test sequences and evaluated under designed conditions. The framework enables systematic exploration of execution behavior and supports the derivation of reliability from observable system evidence.

3.1 Architecture Overview

Figure 2 illustrates the architecture of the proposed framework. The system integrates structured test generation, execution under realistic conditions, and multi-layer observability to analyze system behavior as it occurs in operation.

The framework operates as a pipeline:

$$\Phi(p) \rightarrow \text{Test Generation } (B,\ \beta_k) \rightarrow \text{Execution} \rightarrow \text{Observability} \rightarrow \text{Analysis} \rightarrow \text{Derived Assurance}$$

Here, $\Phi(p)$ denotes the structured prompt specification describing the intended behavior of the system. Test generation produces minimal valid units B and composes them into execution sequences β_k, representing multi-step workflows exercised during evaluation.

Each stage systematically exposes failure modes and enables the derivation of reliability characteristics.

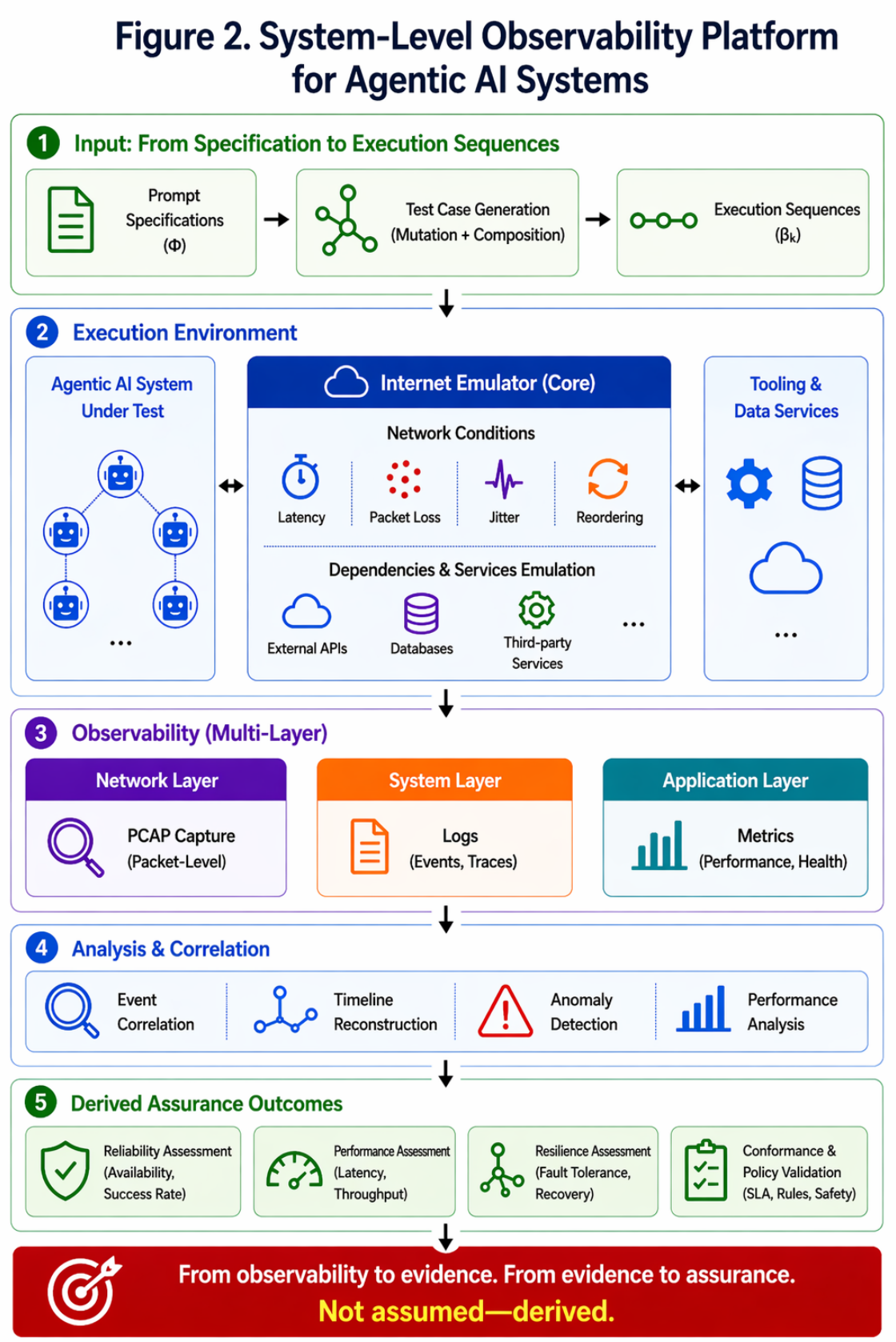

Figure 2: System-level observability platform for agentic AI systems. Structured prompt specifications are transformed into execution sequences β_k and executed under realistic conditions within an Internet Emulator. Multi-layer observability (PCAP, logs, metrics) enables analysis and the derivation of measurable assurance outcomes from observable system behavior.

3.2 Test Generation and Execution Sequences

The process begins with structured prompt specification $\Phi(p)$, which defines the intended behavior of the system. Using mutation-based techniques, minimal valid units B are extracted and used to construct execution sequences β_k.

These sequences represent multi-step workflows that include model inference, tool invocation, data retrieval, and state updates. Both nominal sequences and failure-inducing probe sequences are generated to explore system behavior under diverse conditions.

Execution sequences β_k serve as the fundamental unit of evaluation. Unlike isolated prompt-response interactions, they capture temporal dependencies, interactions across components, and the cumulative effects of execution.

Critically, execution at this level exposes both interaction-induced failures and latent model-level weaknesses that remain hidden under isolated testing.

3.3 Execution Under Realistic Conditions

The generated sequences are executed within an Internet Emulator that reproduces realistic network and system conditions, including latency, jitter, packet loss, dependency variability, and controlled failure injection [11].

Conventional evaluation environments assume idealized conditions and therefore fail to capture the operational dynamics of deployed systems. In contrast, the emulator enables controlled experimentation under degraded and dynamic conditions, reflecting real-world execution.

This capability is essential for exposing failure modes that arise from timing, resource contention, and dependency interactions, which are not observable under idealized evaluation.

3.4 Multi-Layer Observability

System behavior is captured through multiple layers of observability:

- **Packet-Level Observation (PCAP):** Captures protocol-level behavior, including packet exchanges, retransmissions, and communication patterns between system components, enabling the derivation of execution metrics such as latency and delay.
- **System Logs:** Record execution events across services, enabling tracing of workflows and identification of failure points.
- **Metrics:** Provide quantitative measures of latency, throughput, error rates, and resource utilization.

These layers provide complementary visibility into system behavior. In particular, packet-level observation reveals execution dynamics that are not visible through application-level instrumentation alone.

3.5 From Observability to Derived Assurance

The collected observability data is analyzed to derive assurance outcomes. These include:

- Reliability characterization under varying conditions
- Identification of failure modes and triggering conditions
- Coverage of execution sequences and interaction patterns

- Detection of latent model-level weaknesses exposed through execution

Assurance is therefore not assumed from model performance but derived from observable system behavior under realistic execution conditions.

This represents a shift from model-centric evaluation to system-level validation, where reliability is established through evidence gathered during execution.

3.6 Traceability, Explainability, and Independent Validation

In operational settings, a critical requirement is the ability to answer the question: how did the system arrive at a specific conclusion? For agentic AI systems that execute multi-step workflows across models, tools, and services, such traceability cannot be established from model outputs alone.

Traceability refers to the ability to reconstruct the execution pathway that produces an observed outcome, linking system behavior to the sequence of interactions, component invocations, and runtime conditions involved [9]. Using multi-layer telemetry, including logs, traces, and packet-level observations, the framework enables reverse-engineering of observed behavior into a system-level execution pathway that explains how a result is produced.

Beyond explanation, the framework supports independent validation of observed behavior. Reconstructed execution pathways can be re-executed under controlled conditions using structured test generation and an Internet Emulator, allowing verification that the same conditions reproduce the observed outcome.

This combination of traceability and controlled re-execution provides a principled basis for explaining and independently validating how outcomes are produced in agentic AI systems, supporting requirements for reliability, accountability, and evidentiary analysis in operational environments.

3.7 Alignment with Operational Requirements

The framework directly addresses the reliability challenges identified in Section 2. By evaluating execution sequences under realistic conditions and providing visibility into system behavior, it captures the conditions under which failures arise in practice.

In particular, the framework enables analysis of concurrency effects, dependency interactions, and timing variability, all of which are central to operational reliability but absent from conventional evaluation pipelines.

In contrast to model-centric evaluation, the proposed framework enables system-level validation grounded in execution evidence, directly addressing the reliability challenges identified in Section 2.

4 Conclusion

This paper addresses a fundamental gap between evaluation and operational behavior in agentic AI systems. While current practices emphasize model-level evaluation under controlled conditions, real-world evidence shows that failures in deployed systems arise during execution and manifest at the system level. These failures are driven by interactions among components, dependency coupling, concurrency, and variability in runtime conditions.

A central finding of this work is that model-level correctness does not guarantee system-level reliability. Certain classes of errors remain hidden under isolated prompt-response testing and become observable only when systems are exercised through multi-step execution sequences. As a result, evaluation methodologies that do not capture execution behavior and system interactions provide an incomplete basis for assessing operational readiness.

To address this gap, we presented a system-level observability framework for evaluating agentic AI systems under realistic operating conditions. The framework combines structured test generation, execution of multi-step workflows, and multi-layer observability to analyze system behavior as it occurs in operation. By transforming prompt specifications into execution sequences and evaluating them under controlled yet realistic conditions, the approach enables systematic exposure of both interaction-induced failures and latent model-level weaknesses.

Beyond observation, the framework supports traceability and independent validation by reconstructing execution pathways that explain how outcomes are produced and verifying them through controlled re-execution. This enables reliability to be derived from observable system behavior under controlled execution conditions rather than inferred from model performance alone.

The results establish a clear requirement for the reliable deployment of agentic AI systems: evaluation must operate at the level of execution sequences, system interactions, and realistic runtime conditions. Without such evaluation, there is no principled basis for establishing reliability in environments where failure is unacceptable.

More broadly, this work positions system-level observability, traceability, and validation as a foundational layer in the AI evaluation stack. As agentic AI systems are deployed in increasingly critical roles, the ability to generate, observe, explain, and independently validate system behavior will be essential.
From observability to evidence. From evidence to assurance—derived, not assumed.

References

[1] S. Yao *et al.*, "React: Synergizing reasoning and acting in language models," *arXiv preprint arXiv:2210.03629*, 2023.

[2] T. Schick *et al.*, "Toolformer: Language models can teach themselves to use tools," *arXiv preprint arXiv:2302.04761*, 2023.

[3] Y. Chang *et al.*, "A survey on evaluation of large language models," *arXiv preprint*, 2024.

[4] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, *et al.*, "Language models are few-shot learners," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020.

[5] "Openai status page." `https://status.openai.com`, 2024.

[6] "Anthropic status page." `https://status.anthropic.com`, 2024.

[7] "Google cloud status dashboard." `https://status.cloud.google.com`, 2024.

[8] "Aws service health dashboard." `https://health.aws.amazon.com`, 2024.

[9] B. H. Sigelman *et al.*, "Dapper, a large-scale distributed systems tracing infrastructure," in *Google Technical Report*, 2010.

[10] P. Barham *et al.*, "Magpie: Online modelling and performance-aware systems," in *HotOS*, 2004.

[11] D. E. Comer, *Internetworking with TCP/IP*. Pearson, 2018.