Frontiers in Computer Science and Artificial Intelligence
ISSN: 2978-8048
DOI: 10.32996/fcsai
Journal Homepage: www.al-kindipublisher.com/index.php/fcsai

FCSAI
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

| RESEARCH ARTICLE

Observability for LLM apps: what to log, privacy-safe telemetry, KPIs

Prasad Maderamitla[1] and Subba Rao Katragadda[2]
[12]Independent researcher, California, USA
Corresponding Author: Prasad Maderamitla**, E-mail**: prasad.madera@gmail.com

| ABSTRACT

Large Language Model (LLM) applications increasingly form an integral part of enterprise software architecture, enabling conversational interfaces, intelligent assistant applications, and autonomous decision-support systems. While these applications provide tremendous flexibility and capability, their probabilistic nature, prompt dependency, and complex orchestration pipelines create new challenges for monitoring and reliability engineering. The traditional approach to observability, relying on logs, metrics, and traces, is found to be inadequate to measure semantic correctness, behavioral consistency, and governance risks associated with LLM applications. This study explores the concept of observability in large language model (LLM) applications from three different viewpoints: auditable data selection, privacy-preserving telemetry construction, and meaningful operational key performance indicator (KPI) definition. Following the best practices of software observability and MLOps, the study proposes a conceptual framework for model-agnostic observability in LLMs that covers the interaction layer, execution layer, performance layer, and safety layer. In particular, the study focuses on the application of privacy by design, including metadata-centric logging, selective redaction, and controlled access to telemetry data. Furthermore, this paper introduces a well-defined set of operational key performance indicators (KPIs) specific to large language model (LLM) applications, including reliability, performance efficiency, measures of output quality, and safety compliance. The above-mentioned parts of the framework enable the development of a well-structured framework for detecting faults, managing costs, as well as ensuring the reliability of LLMs. The above-mentioned framework makes it easier to implement LLMs at the enterprise level.

| KEYWORDS

Large Language Models (LLMs), Observability, Privacy-Safe Telemetry, Operational Key Performance Indicators (KPIs), AI System Governance

| ARTICLE INFORMATION

ACCEPTED: 01 February 2026 **PUBLISHED:** 14 February 2026 **DOI:** 10.32996/jcsts.2026.5.4.2

1. Introduction

Large Language Models (LLMs) have quickly moved from being purely experimental constructs to becoming integral parts of production-grade software systems. Today's applications increasingly feature LLMs as conversational partners, business copilots, decision-support tools, and autonomous execution engines. Unlike traditional software constructs, LLM-based applications are characterized by non-deterministic behavior, prompt dependency, and complex execution flows that include the invocation of other tools, retrieval systems, and orchestration logic. These characteristics of LLMs are major stumbling blocks for traditional approaches to monitoring and reliability engineering. In traditional software systems, observability is often achieved through well-understood logs, metrics, and traces that allow system engineers to understand the internal state of the system from the outside. However, these approaches are insufficient for LLM-based applications. LLMs are characterized by non-deterministic behavior that is often difficult to understand. A response from an LLM may be syntactically correct but semantically incorrect, i.e., incorrect with respect to the intent of the user [1].

Concomitantly, the unmitigated escalation of the depth of logging poses significant risks. In fact, Large Language Model (LLM) applications often need to manage user data, confidential information, and regulated content. This raises significant concerns with regard to privacy, security, and compliance. The collection of entire prompts and responses for debugging may violate organizational policies and/or legal regulations, especially in industries like healthcare, finance, and government. Thus, the concept of observability for LLM applications needs to strike a balance between depth and privacy [2].

This work attempts to address all the aforementioned challenges by proposing the concept of observability for LLM applications as a multidimensional problem that includes (i) what to log, (ii) how to collect telemetry while being safe, and (iii) how to define meaningful operational Key Performance Indicators (KPIs). This study proposes a model-agnostic framework for observability instead of focusing on particular tools and/or vendor solutions. The main contribution of this work is to propose a systematic methodology for logging, telemetry, and defining meaningful KPIs to ensure reliable, auditable, and trustworthy LLM applications.

2. Literature Review

It has been recognized for some time now that observability is one of the key pillars for developing reliable and scalable software systems. In the context of traditional software systems, the concept of observability has been achieved through structured logging, metrics collection, and distributed tracing, which allows for the characterization of software system behavior without requiring direct access to internal software states. These best practices have been widely adopted for cloud-native software systems and microservices, where the determinism of execution flow and failure allows for the characterization of performance degradation and system faults within reasonable precision.

With the increased adoption of machine learning in software systems, the concept of observability has now been extended to MLOps. In prior studies and best practices, emphasis has been given to the monitoring of data pipelines, model training, inference, and model performance metrics such as accuracy, precision, and recall. In addition, emphasis has also been given to data drift, concept drift, and model degradation over time. However, these approaches have their own strengths and limitations, and they are primarily applicable to supervised learning models where ground truth can be measured and inference behavior is reproducible, which may not be the case for large language models [3].

Some of the current research on LLM-based systems highlights the problem of "emergent observability issues" associated with the generative and probabilistic nature of these models. The increasing number of studies and industry reports highlights the importance of prompt management, token tracking, cost monitoring, and response safety evaluation. The latest research on these systems incorporates issues of hallucination detection, alignment monitoring, and moderation as part of the operational monitoring of these systems. However, this research often lacks cohesion and tool-centric approaches, focusing primarily on individual monitoring capabilities rather than a cohesive approach to observability [4].

Another major aspect of the current research on LLM-based systems is the trade-off between transparency and privacy. Several studies have cautioned against the indiscriminate tracking of prompts and model outputs owing to the potential exposure of personally identifiable information, sensitive business data, or content subject to regulatory compliance. While approaches such as anonymization, partial redaction, and access-controlled logging have been suggested as potential countermeasures, these approaches are typically discussed as compliance-oriented enhancements and not as fundamental principles of the design of an observability architecture [5][6].

From this literature, it can be seen that there exists a significant gap between traditional observability models and the needs of LLM-based applications. While individual components such as cost tracking, safety monitoring, and prompt evaluation have been considered, there is a lack of clear guidance on what represents sufficient and privacy-aware LLM-based observability. The lack of such guidance highlights the need for the development of a systematic and enterprise-focused approach to LLM-based observability [7][8].

3. Methodology

The current study uses a conceptual and design-oriented methodology to address the issue of observability in large language model applications. Given the pace at which large language model platforms are evolving and the absence of standardized evaluation datasets to measure observability in production environments, this methodology is more focused on developing a model-agnostic and tool-neutral framework instead of comparing its performance. The methodology draws upon best practices in software observability, MLOps, and current LLM deployment models to create a framework.

3.1 Conceptual Observability Architecture for LLM Applications

The proposed observability methodology models an LLM application as a multi-strata system rather than a single invocation of inference. In this regard, observability signals are constructed across several connected strata or layers: user interaction layer, prompt orchestration layer, model inference layer, tool and retrieval execution layer, and response post-processing layer. In this sense, the proposed methodology enables a detailed view of where failures, degradations, or misalignments occur without having to inspect sensitive content[9].

3.2 Structured Logging Categories

The methodology proposes four major categories of logs that must be implemented to achieve effective observability for large language models (LLMs). Interaction logs record metadata about user requests and system responses, including templates, response length, and confidence, but exclude actual content. Execution logs describe the internal decision-making process, including tool calls, retrieval steps, and branching decisions at the agent level. Performance logs, on the other hand, focus on efficiency, tracking latency at each step in the pipeline, token usage, and cost allocation. Finally, safety and governance logs monitor policy enforcement, moderation, and policy violations.

3.3 Privacy-Safe Telemetry Design

Another methodological guideline of the framework is the concept of privacy by design telemetry. The framework is based on the importance of metadata-centric logging, timely hashing, redaction, and aggregation. Access to the telemetry data is managed through role-based access controls. This helps to ensure that the observability data is utilized for engineering and governance purposes within the constraints of the data protection laws.

3.4 Methodological Scope and Applicability

Significantly, the proposed methodology is scalable and extensible to cover a wide range of applications, from applications involving a single model to agent-based applications. The proposed methodology can be applied to any domain where LLMs operate under stringent reliability, safety, and compliance constraints, thereby providing a basis for a foundational observability framework for LLM applications.

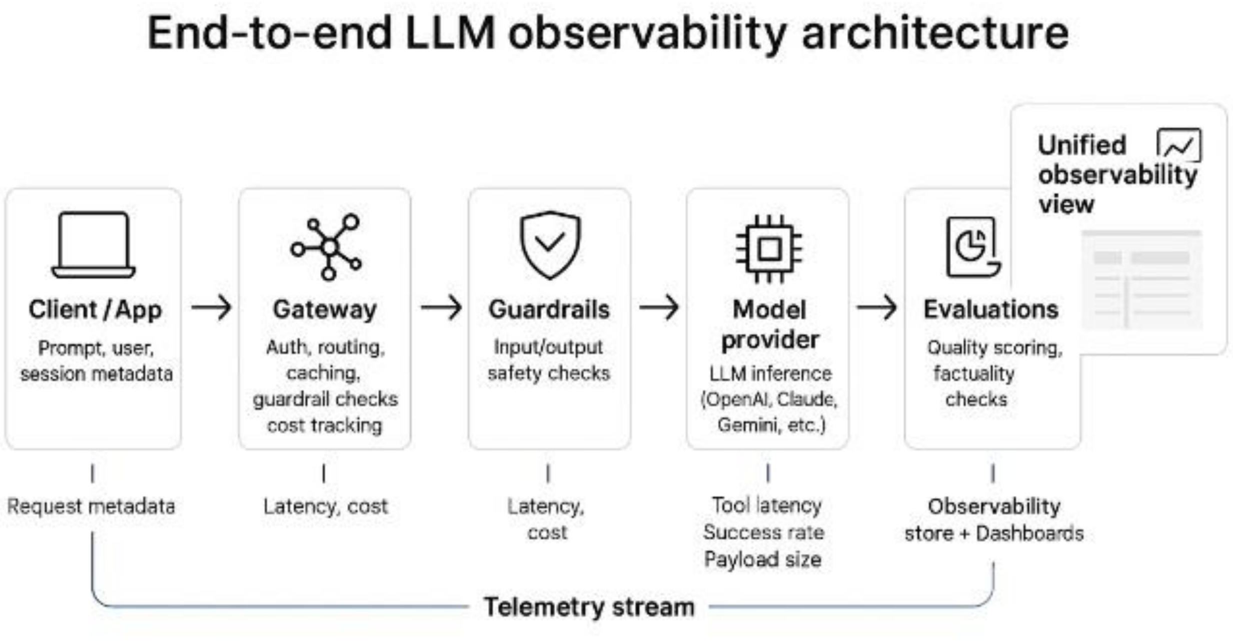
End-to-end LLM observability architecture
Unified observability view
Client / App
Prompt, user, session metadata
Gateway
Auth, routing, caching, guardrail checks cost tracking
Guardrails
Input/output safety checks
Model provider
LLM inference (OpenAI, Claude, Gemini, etc.)
Evaluations
Quality scoring, factuality checks
Request metadata
Latency, cost
Latency, cost
Tool latency Success rate Payload size
Observability store + Dashboards
Telemetry stream

Fig 1. End-to-end LLM observability architecture

The following is a diagram of an end-to-end LLM observability architecture, which integrates each phase of the LLM application pipeline, including the client/application, gateway, safety guardrails, model provider, and evaluation layer, such that all of these components continuously emit signals to a single view of observability. The user request and session information is collected at the client layer, while the gateway layer collects operational signals such as authentication, routing decisions, caching, latency, and cost. The guardrails layer collects safety-related signals through input and output checks, while the model provider layer collects tool latency, success rate, and payload size at the inference level. Finally, the evaluation layer collects quality-related signals such as quality and factuality at a high level. All these diverse signals are combined to form a single view of telemetry and can be visualized through a single view using dashboards, allowing for the monitoring of the entire lifecycle of the LLM application.

4. Discussion

The proposed framework for observability changes the focus of large language model (LLM) applications from the traditional emphasis on the performance and availability of the system to a more holistic approach that is aware of the behavior and the governance of the system. LLMs have the unique characteristic that the output of the system may be operationally correct but functionally incorrect, misleading, or even dangerous. As a result, the proposed framework for observability considers not just the speed and rate of the output but the degree to which the output is aligned with the intent of the user, the policies of the organization, and the cost considerations.

A key outcome of the proposed framework is the definition of the operational Key Performance Indicators (KPIs) that are specific to the large language model applications. The KPIs for reliability include the stability of the execution of the system, the success rate of the invocation of tools in the workflows of the agent, and the rate of retries in the workflow. These KPIs allow the detection of the weaknesses in the system that may be in the orchestration logic rather than the model. The performance KPIs extend the focus on the speed of the output to the efficiency of the utilization of tokens and the cost per successful interaction. These factors are critical in the large language model applications due to the cost of inference.

However, the quality-related KPIs remain inherently difficult to address due to the subjective nature of language generation. Nevertheless, re-prompt frequency, user correction rates, and response consistency on semantically related inputs offer a useful surrogate metric that provides insight into the degradation/misalignment process. This is particularly useful when combined with a lightweight human-in-the-loop feedback process.

The final dimension of observability is related to the KPIs that fall under the safety and compliance aspect. These KPIs include policy violation rates, moderation trigger frequency, and sensitive content exposure rates. These KPIs allow organizations to proactively manage risk. Most notably, the KPIs enable the achievement of auditability and governance requirements, thereby allowing the deployment of LLM applications in a regulated setting.

The discussion also helps clarify the trade-offs. Increased observability improves diagnosability but also increases the amount of data, the cost, and the privacy risks. On the other hand, over constrained telemetry can hide critical failure modes. The proposed framework highlights the need for balanced design decisions, where the level of observability is aligned with the criticality of the business context. These considerations clearly reveal the importance of the observability of large language models as not just a technical problem but a strategic enabler for trust, scalability, and the sustainability of AI systems.

5. Conclusion

As large language models evolve from being merely experimental tools to being an essential part of the overall software architecture for various enterprises, the concept of observability emerges as a key facilitator for reliability, trust, and governance. Inadequacies associated with conventional approaches to monitoring, which are specifically designed for deterministic software and conventional machine learning models, are addressed by proposing a well-defined framework for the overall concept of observability, including aspects related to logging, telemetry, and KPI-based monitoring.

The original contribution is to view the observability of large language model (LLM) applications as a multidimensional design issue rather than a simple accumulation of different tooling features. By defining the scope of the data to be monitored along different strata such as interaction, execution, performance, and governance, and by integrating privacy by design considerations into the monitoring pipeline, the proposed framework enables meaningful observability while ensuring compliance and user trust. The addition of operational key performance indicators (KPIs) translates observability into meaningful insights for different engineering, governance, and business stakeholders.

In conclusion, observability is a key enabler of the successful deployment of large language model systems. Potential future work on the proposed framework could include the development of standard evaluation benchmarks, automatic quality assessment methodologies, and the integration of human-in-the-loop oversight mechanisms to enhance observability.

References

1. Moshkovich D, Zeltyn S (2025) Taming Uncertainty via Automation: Observing, Analyzing, and Optimizing Agentic AI Systems
2. Shanmugarasa Y, Ding M, Arachchige CM, Rakotoarivelo T (2025) SoK: The Privacy Paradox of Large Language Models: Advancements, Privacy Risks, and Mitigation. In: Proceedings of the ACM Conference on Computer and Communications Security
3. Eken B, Pallewatta S, Tran N, et al (2025) A Multivocal Review of MLOps Practices, Challenges and Open Issues. ACM Comput Surv 58:. https://doi.org/10.1145/3747346
4. Anwar U, Saparov A, Rando J, et al (2024) Foundational Challenges in Assuring Alignment and Safety of Large Language Models. Transactions on Machine Learning Research 2024:
5. Kibriya H, Khan WZ, Siddiqa A, Khan MK (2024) Privacy issues in Large Language Models: A survey. Computers and Electrical Engineering 120:. https://doi.org/10.1016/j.compeleceng.2024.109698
6. Mireshghallah N, Li T (2025) Position: Privacy Is Not Just Memorization!
7. Chan A, Ezell C, Kaufmann M, et al (2024) Visibility into AI Agents. In: 2024 ACM Conference on Fairness, Accountability, and Transparency, FAccT 2024
8. Subba Rao Katragadda. (2026). Utilizing LLM models for advanced automation, manufacturing operations. *Journal of Mechanical, Civil and Industrial Engineering*, *7*(2), 08-14. https://doi.org/10.32996/jmcie.2026.7.2.1
9. Anderljung M, Smith ET, O'Brien J, et al (2023) Towards Publicly Accountable Frontier LLMs: Building an External Scrutiny Ecosystem under the ASPIRE Framework