iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

LLM OBSERVABILITY: A FRAMEWORK FOR PRODUCTION APPLICATIONS

Akhil Reddy Mandadi
Independent Researcher, India

ABSTRACT
The Large Language Model (LLM) applications have transitions from experimental prototype systems to production-critical systems that can be used in enterprise automation, intelligent customer services, software engineering, healthcare, manufacturing, and business decision making. Traditional observability techniques treating software systems as metrics, distributed trails, and structured logs are important but lack insight into the distinctive context-specific attributes of the LLM enabled applications. When new requirements arise, such as prompt evolution, model versioning, retrieval quality, tool invocation, output reliability, and token usage, alongside behavioral consistency, for modern LLMs, they all add to the observability challenges. At the same time, they suffer unique operational problems, such as hallucinations, grounding drift, degradation during retrieval, increasing inference costs and performance variability due to model changes. Many technologies have come along since then to solve these problems, most of them commercial or open source, but none of them provide consistent implementation or contextually evaluate all platforms. A four-sheet model that captures production observability for LLMs, ranging from prompt and model provenance, to retrieval and tool call observability, to output quality signals to cost and latency observability. The framework brings together existing engineering practices and functions within a cohesive architecture that can identify critical observability metrics, identify existing capabilities of tools, and determine infra gaps that require resolution. The proposed framework offers practical advice for engineering teams deploying production LLM applications, along with an avenue for future research to bridge toward scalable, transparent, and reliable LLM operations, by introducing baseline observability layers and principles of operations monitoring.
Keywords: Large Language Models (LLMs), LLM Observability, Production AI, AI Monitoring, Prompt Engineering, AI Operations (AIOps), Telemetry, Model Governance.

I. INTRODUCTION

1.1 Background
Artificial intelligence (AI) systems are fast evolving from task-specific predictive tools to generalized tools that reason, produce natural language, communicate with external tools and make autonomous decisions. In the contemporary landscape of digital transformation, companies in numerous industries like healthcare, finance, education, manufacturing, software, and industrial automation are increasingly leveraging LLM based applications as essential tools to tackle intricate workflows, customer interaction, and boost efficiency in various areas. In the early stages, these systems are being deployed in a small number of settings, and ensuring their sustainability during the transition to operational environments is now a critical engineering need and not just an afterthought (Hou et al., 2025).
The three pillars of traditional Software Observability are metrics, logs and distributed traces. These observability primitives offer a method for capturing visibility into the behavior of infrastructure and help engineers perform monitoring, failure diagnosis, and behavior optimization of their application. But production LLM applications are particularly unique compared to traditional software applications since their outputs are probabilistic, exist in multiple versions, can be triggered by specific prompts, heavily dependent and influenced by information retrieved from the database, and are further impacted by external tool interactions. This thus makes it impossible to explain operational behaviours using conventional telemetry alone, which is common (Kamath et al., 2024).
As new applications of LLM become available, there are completely new kinds of operational signals. To analyze and comprehend the performance of applications, engineers need to keep track of prompt evolution, model deployment lineage, retrieval quality, structured output validation, user feedback, token consumption, invocation latency of the tools, and inference costs. Moreover, there has to be ongoing consistency checking of behavior in a production environment because models and prompts change over time, as do databases of supporting knowledge. Requirements for observability go beyond the monitoring of infrastructure to the

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

application intelligence monitoring which requires special instrumentation to capture the semantic and behavioral attributes of the applications along with traditional performance metrics (Chen et al., 2025).

Comprehensive observability becomes more relevant as a result of the complexity of LLM production ecosystems that continue to pile up. As retrieval-augmented generation, multi-agent collaboration, external APIs, cloud services, and domain specific knowledge repositories increasingly become part of a single End-to-End AI workflow, enterprise applications are evolving more and more into such a solution. These interdependencies intertwine and act in a dynamic manner, whereby a failure encountered in one subsystem can propagate itself through multiple services before eventually being captured by the operators. Ensuring that production LLM applications are reliable, transparent, governed and operationally trustworthy requires effective observability as key piece of the puzzle throughout the entire production lifecycle of LLM applications (Dong et al., 2024).

1.2 Challenges of Observing Production LLM Applications

Production LLM apps are not necessarily a breeze to observe as there are significant difficulties to consider, which are different than software systems. Traditional application performance monitoring tools measure activity in the infrastructure, response time for requests, resource utilization and service response availability. Though useful, they do not tell much about whether an LLM gives factually correct answers, handles the prompts correctly, has retrieved relevant knowledge, or has selected appropriate tools for the downstream execution. Therefore, normal infrastructures can fail and not be detected.

The probabilistic aspect of language generation is one of the major issues. Unlike deterministic software which always returns the same output when given the same input, LLM applications can return different outputs with similar operational conditions. Such variation makes it difficult to assess their performance, since parameters other than effectiveness like semantic correctness, grounding to the context and user satisfaction in addition to computational efficiency can affect the quality of a system. A trend in recent research is that in order to get a complete picture of different operational states of LLM applications, with the help of telemetry you had to add behavioral signals (Moshkovich et al., 2025).

There are further challenges with retrieval augmented generation, external tool integration and multi-agent workflows. Production applications often pull data from dynamic knowledge stores, call external APIs and orchestrate communication between several intelligent entities. This is because it could come from stale retrieval indexes, incorrect context grounding of the tools, insufficient tool responses, and inconsistent communication between the agents, which would make root cause diagnosis much harder for online applications. As organizations increasingly adopt autonomous AI in various business critical functions (Fournier et al., 2025; Vyas & Mercangoz, 2025), these challenges intensify every day.

The other challenges faced in production monitoring are caused by the variable inference cost, latency, and provider performance, which changes constantly as part of operational governance. Engineers need to balance the quality for the application with the efficiency of the resources used, as well as meeting the organizational service-level requirements and budget. Therefore, other than monitoring infrastructure, integrated monitoring is necessary for technical performance, operational expenses, and behavioral quality to achieve effective observability (Thirumalai, 2025).

1.3 Research Gap

While significant strides in software observability and LLM development have been made, such as those in software observability and LLM engineering, current LLM monitoring solutions are disjointed when it comes to production LLM use cases. While these traditional observability platforms are largely dedicated to systems that have at least one known, fixed state, they don't quite capture prompt evolution, semantic correctness, retrieval effectiveness, or model behavioural change when these factors are important to observe. This suggests that organizations often end up creating their own monitoring pipelines for their deployment, leading to different engineering practices and industry-wide limited standardization in production pipelines (Ganesan, 2024).

1.4 Research Objective

This study's goal is to create a common definition of production LLM observability that builds a framework for the key monitoring layers needed to ensure that today's LLM applications run reliably. Proposed framework separates the observability into four complementary layers prompt and model provenance, retrieval and tool call telemetry, output quality signals, and cost and latency governance.

The framework also aims at creating viable connections between operation level signatures, engineering goals, existing observation platforms and other remaining gaps of the infrastructure. The study aims to combine these elements into a unified framework to offer actionable insights for the deployment of production and enable

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

standardized monitoring, operational diagnostics, governance, and iterative optimization during the lifecycle of LLM applications.

1.5 Contributions

This research work is significant for the emerging area of producing LLM observability in five main regards. To begin with, it introduces a four-layer observability framework tailored to production LLM applications that builds on and adapts the principles of software observability to address the operational nature of generative AI systems. Second, the framework specifies and structures what observability signals are needed to monitor prompt to prompt change, retrieval quality, model behaviour, output reliability, operational cost and latency in an engineering model.

Thirdly, the proposed framework maps between the production monitoring needs and the current observability tools and platforms at the ready, allowing engineering teams to assess their existing instrumentation and recognize any gaps that may need more infrastructure to meet the requirements. Fourth, the study offers hands on input for enterprise LLM deployments on how to take a scalable observability approach that will enhance operational transparency, production diagnostics, governance, and service reliability. Last, the framework provides a conceptual framework to inspire future research into adaptive monitoring, behavioral drift detection, cross platform telemetry standardization and next generation LLM operations, which all contribute to the ongoing development of production AI systems that are reliable and trustworthy.

2. RELATED WORK

2.1 Traditional Software Observability

Traditionally metrics, logs and distributed traces are three complementary telemetry primitives that together provide visibility into the health, performance and expected operational failures of a system. Together these mechanisms have evolved to be the foundation of Application Performance Monitoring (APM), a tool with the objective of helping engineers troubleshoot infrastructure issues, detect service degradation and keep systems operational. As LLM components in production software, researchers have what researchers say that the traditional observability needs to go beyond infrastructure signals and come to include semantic and behavioral signals. Kamath et al. (2024) pointed out that production LLM systems need to be continuously monitored during deployment, whereas Shethiya (2023a, 2023b, 2023c) showed that the use of generative AI in modern software architectures was likely to alter the assumption of traditional monitoring. Similarly, Ganesan (2024) stated that observability with LLM solutions goes beyond conventional diagnostic capabilities, adding smart behavioural analytics.

Table 1 below compares traditional software observability requirements and emerging requirements for LLM observability in order to highlight the differences.

Table 1 Comparison between Traditional Software Observability and LLM Observability

Observability Aspect	Traditional Software	Production LLM Applications
Primary telemetry	Metrics, logs, traces	Metrics, traces, prompts, model versions, retrieval signals
Monitoring objective	Infrastructure health	Behavioral and operational intelligence
Failure detection	Performance degradation	Hallucination, grounding drift, output quality, cost anomalies
Operational scope	Services and infrastructure	Models, prompts, retrieval, tools, governance

Source: Developed by the authors based on Kamath et al. (2024), Ganesan (2024), and Shethiya (2023a, 2023b, 2023c)

Table 1 shows the increased complexity of observability signals needed to monitor LLM applications as compared to traditional software systems, and consequently the need for observability specific monitoring frameworks.

2.2 Existing Research on LLM Monitoring and Observability

There have been several recent studies examining both behavioural and operational, as well as engineering monitoring of production LLM applications. As discussed above, Chen et al. (2025) presented LLM observability design principles focusing on the developer's requirements for monitoring, while Dong et al. (2024) proposed the design of AgentOps for better observability in autonomous LLM agents. Fournier et al.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

(2025) also examined behavioral variation in agentic AI processes, and Moshkovich et al. (2025) asserted that more than just the standard of benchmark scoring is needed to adhere to the effective optimization of agentic systems. The study also explored other aspects like latent-state observability, production testing, evaluation protocols, and real-time performance assessment, underscoring the critical need for robust monitoring during the entire lifecycle of an LLM (Liu et al., 2025; Ma et al., 2025; Mohammadi et al., 2025; Tamanampudi, 2024). Leveraging LLM observability research to map its evolution conceptually is summarized in **Figure 1.**

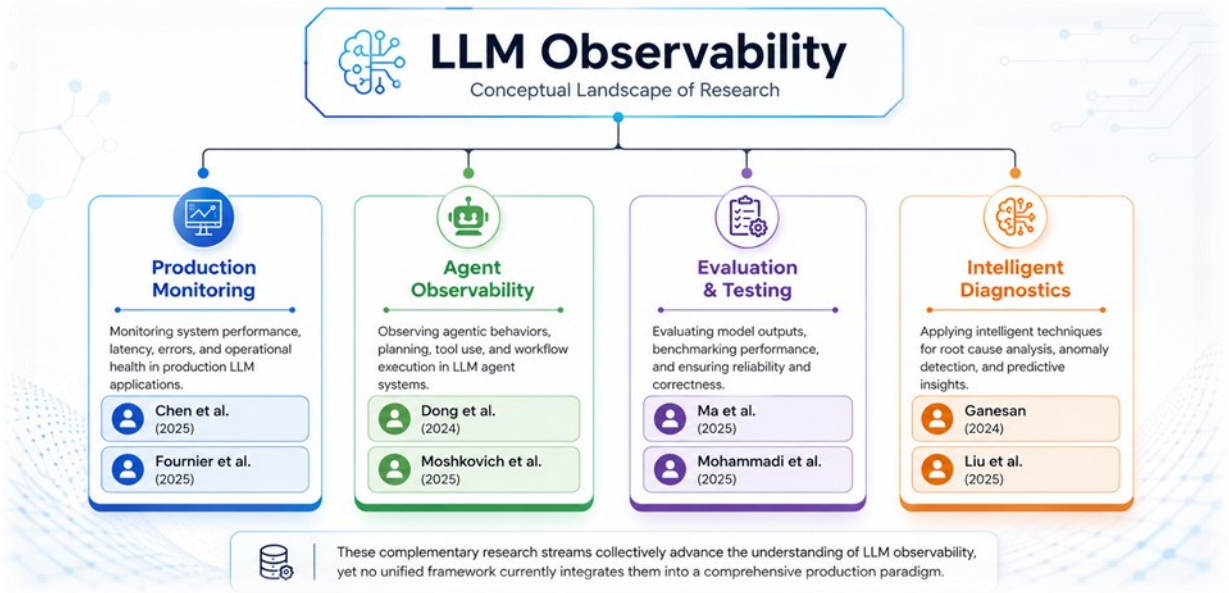

Figure 1. Conceptual landscape of LLM observability research
Source: Developed by the authors based on Chen et al. (2025), Dong et al. (2024), Fournier et al. (2025), Ganesan (2024), Liu et al. (2025), Ma et al. (2025), Mohammadi et al. (2025), and Moshkovich et al. (2025)

As illustrated in **Figure 1**, there are many different themes conducted on each of the complementary subjects that could be addressed in current research, but for the most part, these current studies are individual observability themes and do not provide an integrated production model.

2.3 Current LLM Observability Platforms

LLM applications have been commercialized rapidly, and with that, the creation of specialized observability platforms that can track prompts, model behavior, pipelines (retrieval), and operational performance. With it being a part of intelligent manufacturing, industrial automation, schema integration, and autonomous control, observability has become a core competency of contemporary engineering practices and is now demanded for better deployment reliability (Garcia et al., 2024; Sankiti & Avirneni, 2022; Sivaraman, 2024; Thirumalai, 2025; Vyas & Mercangoz, 2025). Existing platforms often focus on specific features like tracing, evaluation or performance analytics, but lack an overall architecture that enables the integration of provenance, retrieval telemetry, output quality assessment and operational governance. This limitation drives forward the idea of four layers of abstraction for an observability approach, introduced as a framework to streamline a fragmented and less complete observability practice into a complete engineering model for production LLM applications.

3. METHODOLOGY

3.1 Framework Development Process

The proposed LLM observability framework is the result of a systematic synthesis of literature from both the scholarly and production engineering communities and recent research on observability in languages. The methodology was directed at defining what monitoring was required for production LLM applications and associating that monitoring with operational signals that are realistic. LLM deployment, intelligent software architectures, agent observability, and LLM production monitoring, were studied in a systematic way to identify recurring themes in observability (Kamath et al., 2024; Shethiya, 2023a, 2023b, 2023c; Dong et al., 2024). Research activities around applications for manufacturing, agentic systems, evaluation procedures, and

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

behavioral monitoring frameworks (Garcia et al., 2024; Fournier et al., 2025; Mohammadi et al., 2025) were also conducted for additional insights. The framework development stages are summarized in **Figure 2.**

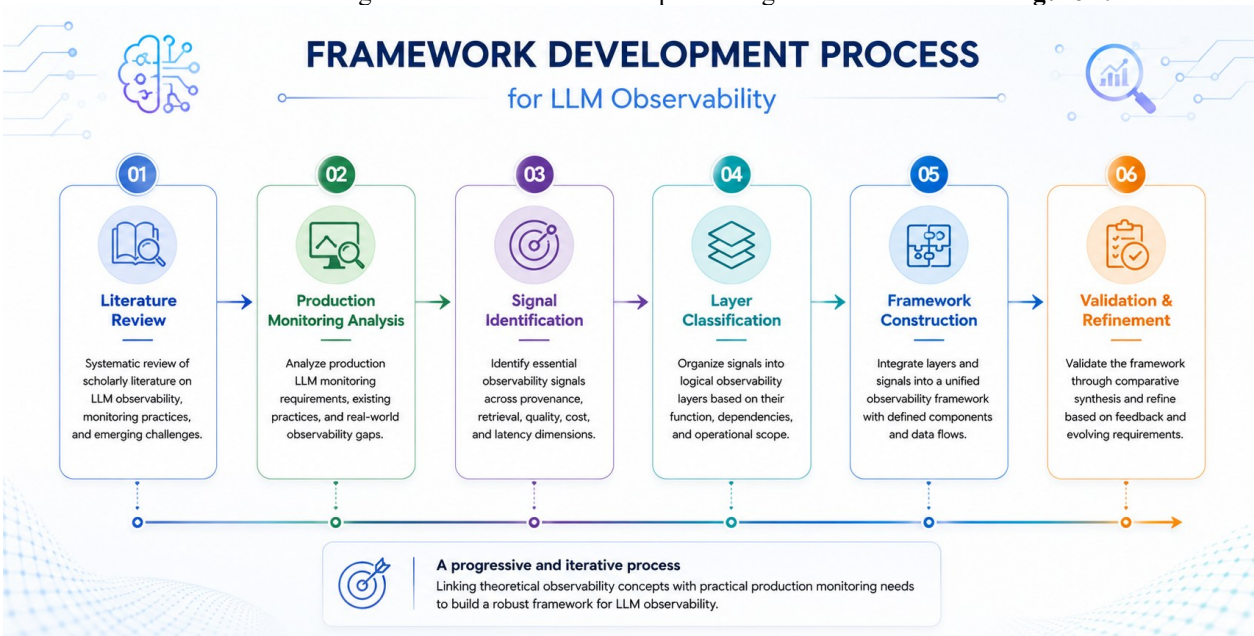

Figure 2. Framework development process for LLM observability.
Source: Developed by the authors based on Dong et al. (2024), Chen et al. (2025), Fournier et al. (2025), and Moshkovich et al. (2025)

Fig. 2 shows how the framework took shape as a step by step process in developing the concepts of theoretically observed observability and the production monitoring requirements.

3.2 Framework Design Criteria

To be useful in practice, the framework was based on production-oriented engineering criteria found in the literature. The criteria for these included effective monitoring, deployment scalability, operational governance, diagnostic capability and observability coverage. The patterns for the framework were chosen based on research on latent-state observability (Liu et al., 2025), production testing (Tamanampudi, 2024; Sivaraman, 2024; Vyas & Mercangoz, 2025), service degradation prediction, and intelligent automation. The design criteria derived are represented in **Table 2.**

Table 2 Framework Design Criteria for Production LLM Observability

Design Criterion	Purpose
Observability Coverage	Capture end-to-end application behavior
Diagnostic Capability	Support root-cause identification
Scalability	Enable deployment across enterprise environments
Governance Support	Monitor compliance, cost, and reliability
Operational Usability	Facilitate engineering decision-making

Source: Developed by the authors based on Chen et al. (2025), Ganesan (2024), Liu et al. (2025), and Kamath et al. (2024)

Table 2 reveals the need for observability to go beyond just collecting telemetry data, encompassing operational diagnostics, governance, and ongoing improvement both in production environments and over time.

3.3 Validation Strategy

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

Comparative synthesis of up to date research work on observability, evaluation, testing, and operation of LLM systems were employed to validate their structures using the method of framework validation. The method of comparative synthesis of current research on the observability and evaluation, testing, and operational management of LLM systems was used to validate their frameworks. To determine if the proposed framework sufficiently addressed underlying observability needs across a broad range of application contexts, studies of the four types of applications discussed above are looked into (Sankiti & Avirneni, 2022; Thirumalai, 2025; Garcia et al., 2024). In addition, recent research on applications paradigms for LLMs, testing methods, and benchmarking frameworks further validated the proposed framework completion and practical applicability (Hou et al., 2025; Ma et al., 2025). The process of validation was used to sharpen the framework so that it is consistent, consistent with its practical use in the production LLM systems, and aligns gradually with their changing needs.

4. PROPOSED FRAMEWORK FOR LLM OBSERVABILITY

4.1 Prompt and Model Provenance

Prompt and model are two basic components of proposed observability framework, as all the downstream behaviors of an LLM application are derived from the interplay between prompts, model configuration, and the versions deployed. Version tracking is critical for ensuring the transparency of production environments, which frequently changes through rapid refinements, models upgrades, parameter tuning and infrastructure changes. If engineers can't determine which events bring about behavior changes without provenance, debugging, rollback and optimizing are complicated. Version management, for example, is cited as a key factor for successful production deployment by Kamath et al. (2024) and Hou et al. (2025). However, Chen et al. (2025) pointed out that developers depend on detailed provenance data to achieve effective LLM observability.

At this layer, the most common observability signals are prompt identifiers, prompts' version history, model version tags, deployment timestamps, configuration information, and execution lineage. These signals can be used together to help engineers rebuild execution histories and discover model drift and behavioural drift through prompts changes. The multi-model, multi-environment deployment of LLM models grows in significance as more and more organizations adopt these services.

4.2 Retrieval and Tool Call Telemetry

Retrieval and Tool calls telemetry are operational execution layer of the proposed framework. Smart LLM apps are integrating retrieval augmented generation, enterprise databases, third party APIs, smart agents, and cloud services more and more. Monitoring just language generation is therefore not enough, as frequently it is the failure of the retrieval pipelines or external tools that leads to failures, not the language model. Dong et al. (2024) showed that LLM agents in these systems need to be visible to each other to interact effectively, and Garcia et al. (2024) showed similar system blindness requirements in intelligent manufacturing systems. **Table 3** summarizes essential telemetry signals and their engineering goals, for operationalization of this layer.

Table 3 Retrieval and Tool Call Telemetry Signals

Observability Signal	Engineering Purpose
Retrieval hit rate	Evaluate knowledge retrieval effectiveness
Retrieval latency	Monitor response efficiency
Tool invocation success	Detect execution reliability
API response latency	Identify external service bottlenecks
Cross-service trace propagation	Support end-to-end diagnostics

Source: Developed by the authors based on Dong et al. (2024), Garcia et al. (2024), Ganesan (2024), and Vyas and Mercangoz (2025)

Operational reliability is a result of retrieval quality and tool execution, as shown in **Table 3.** Collecting telemetry data continuously from these components will help engineers determine whether it is a fault on the infrastructure or a fault in the reasoning/retrieval.

4.3 Output Quality Signals

Beyond infrastructure monitoring, output-quality observability checks if responses are still as expected semantically, operationally, and “functionally”. Production LLM apps have to continually be evaluated on four measures: hallucination frequency, factual consistency, structured output validation, and user satisfaction, as

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

well as behavioral stability. These are all signs of model reliability in real deployments as opposed to controlled benchmark environments. Evaluation, benchmarking, testing, and behavioural analytics have received significant interest and attention in the last few years with a focus on "continuous" evaluation for trustworthy deployment of production LLM systems (Mohammadi et al., 2025; Ma et al., 2025; Moshkovich et al., 2025).

4.4 Cost and Latency Governance

Cost and latency governance give visibility into how resource usage and economic sustainability works. For Production LLM applications, the number of requests often ranges in the millions, leading to varying compute costs dependent on the complexity of prompts, tokens, provider selection, and the number of external tools that are called during the production process. To ensure efficient operations and financial sustainability, observability frameworks should constantly track the inference usage, distribution of response times, service level goals, and budget allocation. Recent studies on intelligent automation, autonomous queries and forecasting service degradation show that cost aware monitoring is becoming an essential engineering requirement to deploy AI in the enterprise (Sivaraman, 2024; Thirumalai, 2025; Ganesan, 2024).

4.5 Cross Layer Integration

The proposed observability framework combines the four monitoring layers and incorporates a process that can be used to support production diagnostics in a comprehensive way. These provide mutually supporting information within the application lifecycle, rather than being separate. For instance, the cause of degraded output quality could be failure to retrieve content as well as additional latency added by external services due to prompt changes. Therefore, cross layer observability allows to get closer to the root cause and to make monitoring more efficient. The four observability layers and how they relate to each other are represented in **Figure 3.**

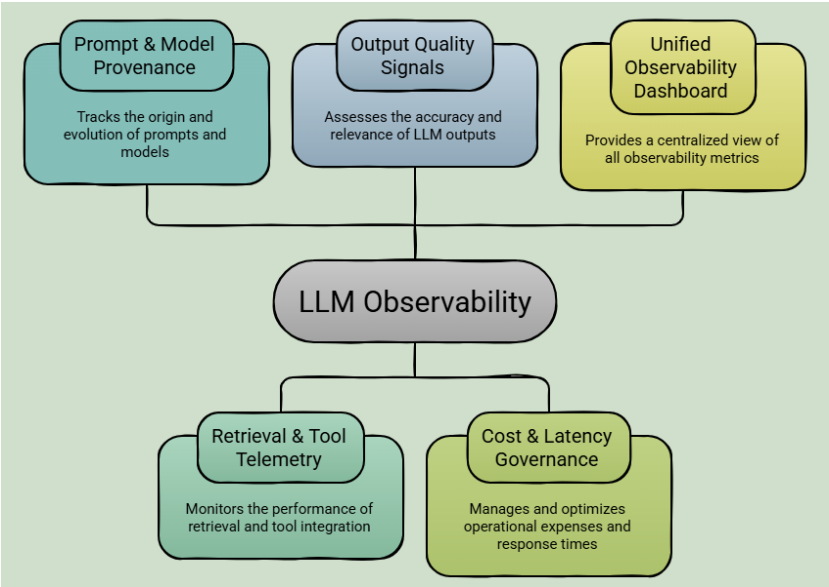

Figure 3. Integrated four layer framework for production LLM observability.
Source: Developed by the authors

Figure 3 shows that in order to achieve good observability, the measurement of the accuracy of the different layers of the operational system must be coordinated, and not limited to individual components of the system. The integrated architecture allows every number of diagnostics, governance and continuous optimization all throughout the production lifecycle.

5. DISCUSSION

5.1 Practical Implications

The proposed framework offers an engineered approach to using production ready observability for LLM applications that are custom to the LLM domain. Provenance, execution telemetry, output quality assessment, and governance are four aspects of monitoring to enable organizations to define a common set of operational dashboards that can aid in incident diagnosis, performance optimization, compliance monitoring, and cost management. The framework also supports complementary deployment strategies of intelligent software

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

systems and industrial automation environment, offering unified monitoring capabilities over heterogeneous infrastructure (Shethiya, 2023a, 2023b, 2023c) and lends itself to synergy with the development of network services as a tool for monitoring the systems (Garcia et al., 2024). It can be summarized by the examples of representative engineering applications of the proposed framework given in **Table 4** below.

Table 4 Practical Applications of the Proposed Observability Framework

Application Area	Expected Engineering Benefit
Production monitoring	Continuous operational visibility
Incident diagnosis	Faster root-cause identification
Model governance	Version control and compliance
Performance optimization	Improved latency and quality management
Cost management	Better resource allocation and budget control

Source: Developed by the authors

As shown in **Table 4,** the proposed framework is based on the extended monitoring that ensures both operational reliability and strategic governance while deploying LLM across enterprises.

5.2 Research Implications

The research approach ensures a uniform conceptual basis upon which future investigations of production LLM observability can be built. Developer Centric Observability, Latent-State Monitoring, Benchmarking Methodologies and Agent Analytics are individually discussed in the existing literature (Chen et al., 2025; Liu et al., 2025; Moshkovich et al., 2025). The suggested framework celebrates these complementary viewpoints by combining them into an engineering model that will help achieve reproducible benchmarking, comparative evaluation, and future observability requirements for production AI.

5.3 Limitations

While the framework brings several benefits, it is purely conceptual and reflects the state of rapidly changing LLM technologies. This study may identify only some of the current requirements for monitoring and does not contain a complete list, as new vendor-specific observability solutions, new model architectures, and an ever continuing shift of deployment practices will add more needs over time. In addition, the effectiveness of the framework needs to be empirically assessed for each of a variety of different industrial contexts, to the extent that it is measurable. Further studies are thus warranted to evaluate its use in large scale enterprise deployments, in addition to improving metrics for observability to support evolving LLM environments.

6. CONCLUSION AND FUTURE WORK

Large language model (LLM) apps are everywhere in enterprise applications, significantly broadening the definition of software observability from its early days focused on infrastructure monitoring. While metrics, logs, and distributed data traces are traditional approaches to observability, they are not sufficient to capture the characteristics of a system in production use of LLM systems, such as their behavior, semantics, and operational context. In this study, these were all addressed with a new four-layer theoretical framework called observability that covers four aspects: prompt and model provenance, retrieval and tool call telemetry, output-quality signals, and cost and latency governance. These complementary layers together offer a framework for intelligent application's behaviour monitoring throughout the entire production lifecycle and help achieve operational transparency, reliability and governance. It unifies the concepts of developer-centric observability, developer-component monitoring, production diagnostics, intelligent software architectures and behavioral analytics, and provides a unified engineering model to tackle some of the most pressing operational problems of current LLM deployments (Chen et al., 2025; Dong et al., 2024; Fournier et al., 2025; Ganesan, 2024; Kamath et al., 2024; Moshkovich et al., 2025).

The proposed framework extends the current research on intelligent software systems, manufacturing applications, agentic AI, latent state observability, testing methodology, and evaluation framework (Garcia et al., 2024; Hou et al., 2025; Kolb & Feigh, 2024; Liu et al., 2025; Ma et al., 2025; Mohammadi et al., 2025). The framework contrasts with existing studies on monitoring and evaluation, focusing on isolated components, offering a broad operating view that enables continuous monitoring and evaluation, performance optimization, governance, and based engineering decision making. In addition, it provides a structure in which observability

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

can be broken down with standard layers of monitoring, which enable easier comparisons across platforms, repeatable research and consistent deployments for different production environments.
The next step is to take the proposed framework to the field with empirical deployment in horizontal workflows and architectures in large scale industrial LLM applications, where the framework can be tested and validated. Further studies are needed to address automated behavioral drift detection, adaptive telemetry generation, intelligent anomaly detection, real-time observability dashboards and standardization of intercropping with observability platforms. Further research is also required to complement the harmonization of the schema, service degradation prediction, autonomous industrial control and cost-aware reasoning in the unified "LLMOps" ecosystems so as to support increasingly autonomous AI applications (Sankiti & Avirneni, 2022; Shethiya, 2023a, 2023b, 2023c; Sivaraman, 2024; Tamanampudi, 2024; Thirumalai, 2025; Vyas & Mercangoz, 2025). As production LLM systems continue to evolve, comprehensive observability frameworks will remain indispensable for ensuring trustworthy, scalable, transparent, and resilient AI-driven software systems.

REFERENCES

1) Chen, X., Li, Y., & Wang, X. (2025, April). Design principles and guidelines for LLM observability: Insights from developers. In *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems* (pp. 1–9). https://doi.org/10.1145/3706599.3719914
2) Dong, L., Lu, Q., & Zhu, L. (2024). AgentOps: Enabling observability of LLM agents. *arXiv preprint arXiv:2411.05285*. https://arxiv.org/abs/2411.05285
3) Fournier, F., Limonad, L., & David, Y. (2025). Agentic AI process observability: Discovering behavioral variability. *arXiv preprint arXiv:2505.20127*. https://arxiv.org/abs/2505.20127
4) Ganesan, P. (2024). LLM-powered observability: Enhancing monitoring and diagnostics. *Journal of Artificial Intelligence, Machine Learning & Data Science, 2*(2), 1329–1336. https://doi.org/10.51219/JAIMLD/premkumar-ganesan/304
5) Garcia, C. I., DiBattista, M. A., Letelier, T. A., Halloran, H. D., & Camelio, J. A. (2024). Framework for LLM applications in manufacturing. *Manufacturing Letters, 41*, 253–263. https://www.sciencedirect.com/science/article/pii/S2213846324000920
6) Hou, X., Zhao, Y., & Wang, H. (2025). LLM applications: Current paradigms and the next frontier. *arXiv preprint arXiv:2503.04596*. https://doi.org/10.48550/arXiv.2503.04596
7) Kamath, U., Keenan, K., Somers, G., & Sorenson, S. (2024). LLMs in production. In *Large Language Models: A Deep Dive*. Springer, Cham. https://doi.org/10.1007/978-3-031-65647-7_8
8) Kolb, J., & Feigh, K. M. (2024, October). Inferring belief states in partially-observable human-robot teams. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 7115–7122). IEEE. https://doi.org/10.1109/IROS58592.2024.10801316
9) Liu, T. Y., Soatto, S., Marchi, M., Chaudhari, P., & Tabuada, P. (2025). *Observability of latent states in generative AI models*. https://openreview.net/forum?id=RaroYIrnbR
10) Ma, W., Yang, Y., Hu, Q., Ying, S., Jin, Z., Du, B., ... & Jiang, L. (2025). Rethinking testing for LLM applications: Characteristics, challenges, and a lightweight interaction protocol. *arXiv preprint arXiv:2508.20737*. https://arxiv.org/abs/2508.20737
11) Mohammadi, M., Li, Y., Lo, J., & Yip, W. (2025, August). Evaluation and benchmarking of LLM agents: A survey. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Vol. 2, pp. 6129–6139). https://doi.org/10.1145/3711896.3736570
12) Moshkovich, D., Mulian, H., Zeltyn, S., Eder, N., Skarbovsky, I., & Abitbol, R. (2025). Beyond black-box benchmarking: Observability, analytics, and optimization of agentic systems. *arXiv preprint arXiv:2503.06745*. https://arxiv.org/abs/2503.06745
13) Sankiti, S. R., & Avirneni, M. (2022). *Safe and scalable data integration using LLMs for schema harmonization and rule synthesis*. https://philpapers.org/archive/SANSAS-13.pdf
14) Shethiya, A. S. (2023). LLM-powered architectures: Designing the next generation of intelligent software systems. *Academia Nexus Journal, 2*(1). https://academianexusjournal.com/index.php/anj/article/view/21/22
15) Shethiya, A. S. (2023). Redefining software architecture: Challenges and strategies for integrating generative AI and LLMs. *Spectrum of Research, 3*(1). https://spectrumofresearch.com/index.php/sr/article/view/22
16) Shethiya, A. S. (2023). Rise of LLM-driven systems: Architecting adaptive software with generative AI. *Spectrum of Research, 3*(2). https://spectrumofresearch.com/index.php/sr/article/view/23

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

17) Sivaraman, H. (2024). *Intelligent automation for service degradation prediction using LLMs and observability*. *International Journal of Core Engineering & Management, 6*(11). https://doi.org/10.5281/zenodo.14342920
18) Tamanampudi, V. M. (2024). Development of real-time evaluation frameworks for large language models (LLMs): Simulating production environments to assess performance stability under variable system loads and usage scenarios. *Distributed Learning and Broad Applications in Scientific Research, 10*, 326–359. https://www.researchgate.net/publication/384804143
19) Thirumalai, V. (2025). *WhatWhen2Ask: Cost-aware LLM querying for autonomous robots in uncertain environments* (Doctoral dissertation, Massachusetts Institute of Technology). https://dspace.mit.edu/entities/publication/a91e47f8-d094-4f2d-9213-fa3d4456e46a
20) Vyas, J., & Mercangoz, M. (2025). Autonomous control leveraging LLMs: An agentic framework for next-generation industrial automation. *arXiv preprint arXiv:2507.07115*. https://arxiv.org/abs/2507.07115

