2026 Volume: 02 Issue: 06 www.ijrpa.com ISSN 2456-9995 Review Article

International Journal Research Publication Analysis

Page: 01-08

BUILDING AN OPEN-SOURCE AI OBSERVABILITY PLATFORM FOR LLM SYSTEMS

***Nayan Patel**

United State.

Article Received: 2 May 2026
Article Revised: 22 May 2026
Published on: 12 June 2026

***Corresponding Author: Nayan Patel**
United State.

DOI: https://doi-doi.org/101555/ijrpa.5856

How to monitor prompts, responses, latency, and hallucinations in production generative AI applications.

INTRODUCTION: The Observability Gap in LLM Systems

Large Language Models (LLMs) are rapidly becoming core components of modern software systems. Applications such as AI copilots, chatbots, automated support systems, and knowledge assistants rely heavily on generative AI models to produce responses in real time.

However, deploying LLM-powered applications introduces a new challenge: observability.

Traditional monitoring tools focus on infrastructure metrics such as CPU utilization, memory consumption, and API latency. While these metrics are essential for system reliability, they fail to provide insight into the behavior and quality of AI-generated responses.

For example, engineering teams often struggle to answer questions like:

- Why did the model generate an incorrect response?
- Which prompts are causing failures?
- How much are we spending on tokens?
- Are hallucinations increasing over time?

To address these questions, organizations are increasingly adopting AI observability platforms such as Langfuse, LangSmith, and Arize AI.

These platforms introduce a new monitoring layer specifically designed for generative AI systems. This article explores how to design and build an open-source AI observability platform for LLM applications.

What Is AI Observability?

AI observability refers to the ability to monitor, analyze, and debug AI model behavior in production environments.

Unlike traditional monitoring, AI observability focuses on the internal dynamics of machine learning systems, including model inputs, outputs, and evaluation metrics.

An AI observability system typically tracks several key layers:

Observability Layer	Purpose
Prompt Observability	Tracks prompts sent to the LLM
Response Observability	Stores generated responses
Performance Observability	Measures latency and throughput
Cost Observability	Tracks token consumption
Quality Observability	Detects hallucinations and response accuracy

Traditional observability answers questions such as “Is the server running?”.

AI observability answers questions like “Is the model generating correct answers?”.

This distinction is critical when deploying generative AI systems in production environments.

Architecture of an AI Observability Platform

An AI observability platform can be implemented as a pipeline that collects telemetry from LLM applications and processes it for monitoring and analysis.

A typical architecture includes the following components:

1. Monitoring SDK
2. Telemetry collector API
3. Event streaming pipeline
4. Storage layer
5. Evaluation engine
6. Observability dashboard

The high-level architecture looks like this:

LLM Application

|

Monitoring SDK

|

Telemetry Collector

|

Event Streaming Platform

|

Processing Engine

|

Observability Database

|

Analytics Dashboard

Several open-source technologies can support this architecture, including:

- Apache Kafka for event streaming
- PostgreSQL for structured telemetry storage
- Grafana for visualization and monitoring dashboards

This architecture allows the system to process telemetry events at scale while maintaining real-time observability.

Capturing LLM Telemetry

The first step in building an observability platform is capturing LLM telemetry data.

Telemetry represents structured events generated when an application interacts with a language model. This data typically includes:

- Prompt text
- Model name
- Generated response
- Token usage
- Request latency

An example telemetry event may look like this:

```
{
 "prompt": "Explain Kubernetes",
 "model": "gpt-4",
 "latency": 1.2,
 "tokens": 324,
```

"response": "Kubernetes is a container orchestration platform..."
}

This telemetry data is captured by a lightweight monitoring SDK integrated into the AI application.

The SDK acts as a middleware layer between the application and the LLM API, ensuring every interaction is logged and analyzed.

Capturing structured telemetry enables engineering teams to debug prompt failures, analyze model performance, and track operational costs.

Monitoring Key LLM Metrics

Once telemetry is collected, the platform must analyze several key metrics that reflect system health and AI performance.

Performance Metrics

Performance metrics help ensure the system remains responsive and reliable.

Important metrics include:

- Response latency
- Request throughput
- Error rates

Latency monitoring is particularly important in user-facing AI applications such as chatbots or copilots.

Cost Metrics

LLM APIs typically charge based on token usage. Monitoring token consumption helps organizations control operational costs.

Cost-related metrics include:

- Total token usage
- API request volume
- Cost per prompt

These metrics allow teams to identify inefficient prompts or excessive API usage.

Quality Metrics

Quality monitoring focuses on evaluating the accuracy and usefulness of generated responses. Common quality metrics include:

- Hallucination rate
- Answer relevance
- Toxicity detection

Evaluation frameworks such as TruLens and Weights & Biases can help automate these evaluations.

Detecting Hallucinations and Quality Issues

One of the biggest challenges with generative AI systems is hallucination — when a model generates plausible but incorrect information.

AI observability platforms use several techniques to detect hallucinations and response quality issues.

LLM-as-a-Judge Evaluation

A secondary LLM evaluates the output of the primary model and assigns a quality score.

Embedding Similarity Analysis

Embedding models compare generated responses against ground-truth answers to measure semantic similarity.

Human Feedback Loops

Users or human reviewers provide feedback on model responses, improving evaluation accuracy over time.

These evaluation pipelines help organizations identify problematic prompts and continuously improve model performance.

Building the Observability Dashboard

Once telemetry and evaluation metrics are processed, they must be visualized through a monitoring dashboard.

A well-designed dashboard provides real-time insights into system behavior and model performance.

Typical dashboard components include:

- Request volume charts
- Latency distribution graphs
- Token consumption trends
- Prompt analytics
- Hallucination score trends

Visualization tools such as Grafana and Metabase are commonly used for building these dashboards.

These dashboards allow engineering teams to quickly detect anomalies and diagnose system issues.

Scaling the Monitoring Platform

As LLM applications grow, the observability platform must handle increasing volumes of telemetry data.

Several techniques can support scalability:

- Distributed telemetry collectors
- Event streaming pipelines
- Batch processing systems
- Containerized microservices

Modern platforms often deploy monitoring services using container technologies such as Docker and orchestration platforms like Kubernetes.

This infrastructure allows observability platforms to process millions of telemetry events per day.

Real-World Use Cases

AI observability platforms are becoming essential in many real-world applications.

Customer Support Chatbots

Monitoring helps identify incorrect responses and improve customer experience.

AI Coding Assistants

Observability tools track prompt performance and response accuracy for developer tools.

Enterprise Knowledge Assistants

Organizations monitor AI responses to ensure accurate and compliant information retrieval.

In all these scenarios, observability ensures reliability, transparency, and accountability for AI systems.

The Future of AI Observability

As generative AI systems become more complex, observability will play a critical role in AI operations.

Several trends are shaping the future of this field:

- Autonomous AI agents
- Multi-model orchestration
- AI governance and compliance frameworks
- Continuous evaluation pipelines

In the coming years, AI observability platforms will become a core component of the MLOps and DevOps ecosystem.

CONCLUSION

Large Language Models introduce a new category of monitoring challenges that traditional observability tools cannot address.

Organizations deploying generative AI systems must monitor not only infrastructure performance but also model behavior, response quality, and operational cost.

By implementing an AI observability platform with telemetry capture, evaluation pipelines, and analytics dashboards, engineering teams gain the visibility needed to operate LLM systems reliably in production.

Open-source tools and modern data pipelines make it possible to build scalable AI observability platforms that empower teams to monitor, evaluate, and continuously improve generative AI applications.