International Journal of Scientific Research and Modern Technology (IJSRMT) *ijsrmt.com*
Volume 3, Issue 4, 2024
DOI: https://doi.org/10.38124/ijsrmt.v3i4.1566

From MLOps to LLMOps: A Position on Engineering Discipline for AI-Native Applications

Akhil Reddy Mandadi[1]

[1]Independent Researcher

Publication Date: 2024/04/28

Abstract

Artificial intelligence (AI) has rapidly become an integral part of the engineering world, moving from traditional software development towards increasingly intelligent, adaptive, and autonomous applications. Machine Learning Operations (MLOps) has evolved as an established engineering approach for deploying, monitoring, and maintaining machine learning systems in production, but the advent of large language models (LLMs) has brought new operational characteristics that go beyond what is currently known in MLOps. LLM-based systems inherently depend on the design of prompts, the external foundation models, the retrieval processes, the contextual information that is needed to be dynamic, and the probabilistic results that cannot always be guaranteed. These properties pose novel engineering problems in terms of lifecycle management, reliability, cost of operations, execution of autonomous activities, and assessment of the non-deterministic responses. This paper proposes to treat LLMOps as a new discipline, not just a new buzzword for MLOps. It provides a conceptual position for LLMOps, explains what it means to incorporate the scope of engineering into LLMOps, and outlines five engineering practices which are essential for the successful operation of LLMOps compared to traditional MLOps: prompt versioning, fidelity grounding, cost control, action authorization, and assessment for non-deterministic outputs. The paper also brings up implications for the future transition of the software engineering research and industrial practice and proposes a research agenda for AI-native applications. By treating LLMOps as a standalone engineering discipline, there is a more solid conceptual basis for AI systems that are reliable, secure, scalable, and economically viable to support more and more autonomous software applications.

Keywords: *LLMOps; MLOps; Large Language Models; AI Engineering; AI-Native Applications; Software Engineering; Prompt Engineering; AI Governance.*

I. INTRODUCTION

AI is no longer just an analytical tool but a core component of contemporary software, having become a specialized technology.AI is now a specialized technology and not only an analytical tool but also a backbone of the software systems of today. AI is now being adopted in various sectors for customer support, software, healthcare, finance, manufacturing, education, and public administration, with applications set to run seamlessly, grow efficiently, and provide consistent results in dynamic settings. With the increasing adoption of AI, the engineering problems faced in deploying and maintaining AI production systems have become much more challenging. These are not only challenges related to building accurate predictive algorithms but relate to the entire AI system lifecycle, from deployment and monitoring to governance, security, reproducibility, and continual improvement.

To overcome these operational problems, Machine Learning Operations (MLops) has been introduced as an engineering discipline, which combines software engineering approaches, DevOps principles, and machine learning processes. Finally, MLOps, by providing standardized methods to automate the deployment of models, manage datasets, monitor model performance, support continuous integration and continuous deployment (CI/CD), and allow for reproducible experimentation across machine learning projects, yielded standardized solutions for these challenges (Kreuzberger et al., 2023). These practices have helped many enterprise deployments of predictive machine learning systems be more reliable and scalable over the past ten years.

Reddy Mandadi, A. (2024). From MLOps to LLMOps: A Position on Engineering Discipline for AI-Native Applications. *International Journal of Scientific Research and Modern Technology*, *3*(4), 71–80. https://doi.org/10.38124/ijsrmt.v3i4.1566

Yet, with the rapid rise of foundation models and large language models (LLMs), the production engineering needs for AI systems have dramatically shifted. The applications that are powered by LLM are different from traditional machine learning models, which rely on structured data and deterministic prediction. They rely on prompt engineering, contextual data, external model providers, retrieval mechanisms, and probabilistic text generation. When it comes to deploying applications built with AI, many of the assumptions that go into traditional MLOps are no longer applicable. Organizations face challenges like managing the lifecycle of their tools promptly, mitigating hallucination issues, and dealing with the cost of operations using tokens, being dependent on external APIs, enforcing security for tools execution, and evaluating responses continuously that can't be measured by traditional accuracy metrics.

This paper presents an argument that these new challenges are more than just an incremental extension of MLOps. Rather, they represent the beginning of a new kind of engineering that has different operating principles, governance structures and software engineering methods. The paper doesn't just see LLMOps as another phase in the MLOps workflow but as a new discipline tailored for the era of foundation models and generative AI, an era where AI is the backbone of new systems.

This is not a position paper on the evolution of MLOps principles to LLM-based systems, but rather a demonstration of the need for new engineering abstractions, lifecycle practices, and evaluation methodologies due to the operational challenges of LLM-based systems. The paper aims at providing a conceptual framework that can inform future software engineering research, industry uptake, and community-led standardization by making LLMOps clear its concept from the traditional MLOps.

- *Why Existing MLOps is No Longer Enough*

While the concepts of MLOps have helped greatly in the operational management of machine learning systems, the underlying principles were originally created for predictive models trained on structured data and run in more controlled settings. It is futile to consider these assumptions when it comes to other emerging applications of LLMs, which operate very differently from traditional machine learning systems.

The effect of prompt is one of the major differences. Traditional MLOps workflows often have "the trained model" as the primary software artifact, and "inference" as the process of applying the learned parameters to new incoming data. Conversely, LLM applications rely on highly crafted prompts that significantly impact the models' output. The prompt modifications can significantly impact the outputs without modifying the model itself, and prompts, like other software, need to be systematically versioned, tested, documented, governed, and managed throughout their lifecycle. Our study on the prompt engineering shows that the structured design of prompts is gaining more and more traction as a key tool for shaping model performance and application behaviour (White et al., 2023).

One other drawback of conventional MLOps is that it involves non-deterministic activity. Typically, the output of a conventional predictive model is the same for the same input. In contrast, LLM's are known to produce varying output for the same input, due to their probabilistic decoding strategies and stochastic generation mechanisms. Therefore, the only metrics used to assess production quality aren't necessarily the standard ones like accuracy, precision or recall. Rather, organization's need to evaluate the quality of reasoning, factual consistency, objectivity, robustness and satisfaction with the output by the user repeatedly over a number of outputs that are generated. Research comparing ChatGPT to humans shows that reasoning ability and hallucination tendency differ significantly in different tasks and domains, emphasizing the difficulty of assessing generative AI systems (Bang et al., 2023).

There needs to be significant redefinition of operational monitoring as well. Current observability approach has mainly been geared towards infrastructure metrics, model latency, prediction quality, and resource utilization. There are, however, several metrics that need to be monitored for production LLM systems, including prompt performance, retrieval quality, and the rate of hallucination, consistency of responses, token usage, and runtime assessment. The recent research on continuous evaluation pipelines underscores the need to move beyond the traditional validation of models to ongoing observability at runtime for LLM-based applications. Likewise, a comparative study of monitoring frameworks has shown that the current methods of observability need major adaptations to meet these new operational facets (Dhanekula & Miah, 2022).

Lastly, as the complexity of AI systems grows, their operational issues are more likely to stem from issues that arise between prompts, external API services, retrieval pipelines, user inputs, and orchestrator frameworks than from defects in the AI model itself. Comprehensive engineering practices throughout the AI lifecycle are crucial, as demonstrated by research showing that machine learning systems often deploy without being aware that there are operational bugs that are silently present (Tambon et al., 2022).

Together, these developments signal that the industrialization of machine learning with MLOps is still essential for traditional ML systems, but not enough to cover the needs of the AI-native applications built with LLMs.

- *Position Statement*

In this paper, we argue that LLMOps is indeed a new discipline of software engineering and not merely a variation of MLOps. LLMOps builds on the core principles of DevOps and MLOps (automation, continuous

deployment, monitoring, collaboration, lifecycle management); however, the engineering artifacts, operational risks, governance requirements, evaluation methods, and architectural premises are very different from more traditional machine learning systems.

LLMOps is all about engineering AI-native applications whose behaviour is the result of the interplay between foundation models, prompts, contextual retrieval, external services, autonomous reasoning, and human oversight. Hence, engineering success is not only about controlling models, but also prompt evolution, grounding fidelity, operational costs and authorizing autonomous action, as well as evaluating prompt outputs continuously during the entire system lifecycle, which is non-deterministic.

By acknowledging LLMOps as a distinct field, it offers a conceptual framework for establishing specific engineering methods, standards, educational programs, benchmarks, and governance models that can help propel the next generation of intelligent software systems.

- *Contributions*

This position paper contributes to the ongoing discussion about engineering AI-native applications in five main ways.

First, it explicitly differentiates between MLOps and LLMOps, showing that the operational considerations of traditional machine learning systems don't necessarily reflect the architectural and engineering reality of foundation model–based systems.

Secondly, by making prompts and contextual grounding, runtime reasoning, external model dependencies, and autonomous interactions first-class operational concerns which demand dedicated lifecycle management, it outlines the engineering scope of LLMOps.

Thirdly, it highlights five practices that make LLMOps different from traditional MLOps: prompt versioning, grounding fidelity, cost governance, action authorization, and evaluation of non-deterministic outputs. These practices are important concepts to grasp in relation to the specific needs of software that is created with an AI-first approach.

Fourth, the paper explores the implications of considering LLMOps as a new engineering discipline for software engineering research and industry. As a result, it emphasizes the need for novel engineering standards, observability, governance, and assessment models that can help support increasingly autonomous AI systems.

Lastly, the author outlines a research research roadmap for the future, prompting further research and development efforts that bring together academia, industry, and the software engineering community to make LLMOps a full-fledged engineering discipline able to solve the new operational challenges found in the use of large language models.

II. FROM MLOPS TO LLMOPS: WHY THE PARADIGM CHANGED

The shift from Machine Learning Operations (MLOps) to Large Language Model Operations (LLMOps) is not just an evolution of current operational methods; it's a natural progression that embodies the transformative power of AI. Rather, it signifies a paradigm shift in the design, implementation, management and ongoing enhancement of artificial intelligence systems. To solve the operational complexity of primarily using structured data, repeatable training processes, and relatively predictable prediction tasks, a new paradigm known as traditional MLOps came into the picture.

Traditional MLOps came into being to tackle the operational complexity of machine learning models that were based mainly on structured data, repeatable training processes, and relatively predictable prediction tasks. By contrast, cutting-edge AI-centric applications increasingly rely on "foundation models “programs that respond to the context, can tap external knowledge bases, call software programs, and reason probabilistically, as opposed to deductively. The architectural shifts have shifted the engineering challenges of production AI systems and added operational challenges that originally weren't part of MLOps. To comprehend this shift, it is important to reflect on the accomplishments of MLOps and the novel engineering challenges have emerged from foundation models.

- *What MLOps Solved*

The need to continuously manage machine learning systems, beyond model development, has come to the fore, leading to the creation of MLOps. Prior to MLOps, companies experienced issues with replicating experiments, dealing with data, data science and software teams, and keeping models deployed as production environments changed. As a result, MLOps integrated machine learning processes with software engineering best practices, such as standardized approaches, to enhance scalability, consistency, and reliability in machine learning operations (Kreuzberger et al., 2023).

This is one of the most impactful aspects of MLOps, as MLOps made it essential to have strong practices in place for managing datasets and building features. Data quality is critical for the success of machine learning models, so organizations turned to features for dataset versioning, feature stores, metadata management, and automated data preprocessing pipelines. These capabilities helped development teams to repeat experiments, and minimized discrepancies between development and production environments (Ruf et al., 2021).

The other significant improvement was in model versioning. Instead of believing the trained models as research products, MLOps developed central model

registries which included information about model metadata, training setups, evaluation metrics, and deployed model histories. This enabled the implementation of the controlled deployment, rollback processes, governance, and regulatory compliance within production systems (Tabassam, 2023).

Similar to continuous integration and continuous deployment (CI/CD) in software development, the concept of CI/CD was applied to the field of machine learning engineering. Automated pipelines allowed to achieve the efficient training, validation, testing, and deployment of models with minimal manual effort. These pipelines enhanced data scientists and software engineers' collaboration by making it reproducible and transparent to operations (Mäkinen et al., 2021), and are often paired with experiment tracking systems.

MLOps also enhanced operational monitoring by providing observability of the infrastructure, performance dashboards, latency analysis, and metrics on resource utilization. More crucially, specialized monitoring features tackled model drift and concept drift by detecting modifications in data distributions, or prediction behavior, over time. Organizations were able to use newly collected data sets to retrain the models, thus maintaining the predictive performance throughout the operational lifecycle, through the use of automated pipelines for retraining (Vangala, 2022). This has led to the emergence of MLOps as a well-developed engineering discipline, one that can help power enterprise-scale predictive AI systems.

- *What Changed with Foundation Models*

While MLOps greatly helped with the operational needs of traditional machine learning systems, the rise of foundation models has revolutionized the engineering landscape. Many modern AI systems connect with external foundation models via application programming interfaces (APIs), while traditional machine learning applications are built using models that are trained within their own systems. This architectural change weakens the organizational control of the underlying models and strengthens the reliance on external providers for whom it is impossible to determine the rates and conditions of their services, as well as their availability of services (Skafidas & Tsoumakas, 2023).

With the growing number of closed-source foundation models, operational management has become even more complex. Many organizations deploy applications with no access to any model architecture and no training sets or training procedures to optimize the application, and so traditional retraining techniques are not feasible. Rather, software engineers are starting to optimize the behavior of the system by fine-tuning their prompts, retrieval strategies, orchestration logic, and contextual information instead of tweaking the models themselves (White et al., 2023).

Another challenge is model evolution at a rate that is fast enough. Model providers regularly update their models with new versions to enhance reasoning capacity and/or extend the context window or change the way they respond. As the models continue to evolve, organizations need to consider compatibility and keep track of the consistency of output, and periodically re-assess operational risks when model versions change. Thus, the lifecycle management does not only apply to artifacts that are controlled by the organization, but also to the continuous evaluation of AI services that are constantly changing in the outside world (Kaddour et al., 2023).

One of the key architectural elements of AI-native applications is also Retrieval-Augmented Generation (RAG). In addition to utilizing pre-trained knowledge, these advanced LLM systems can dynamically fetch information from organizational documents, databases or the web, enhancing the factual accuracy and context relevance of the output. The number of engineering responsibilities has also grown, going beyond traditional MLOps practices, to include retrieval quality, knowledge freshness, grounding fidelity and citation reliability (Skafidas & Tsoumakas, 2023).

Lastly, there has been a considerable increase in the complexity of operations due to the appearance of autonomous AI agents. Agent frameworks allow multiple language models to work together, call external programs, run workflows and interact with enterprise systems without human intervention. These capabilities raise new engineering challenges regarding access control, security, accountability, and governance of operations, not typically faced by the traditional predictive machine learning systems (Wu et al., 2023).

Conceptually, the intellectual evolution of the engineering disciplines which has led to the advent of LLMOps is illustrated in Fig 1 before comparing the two paradigms.

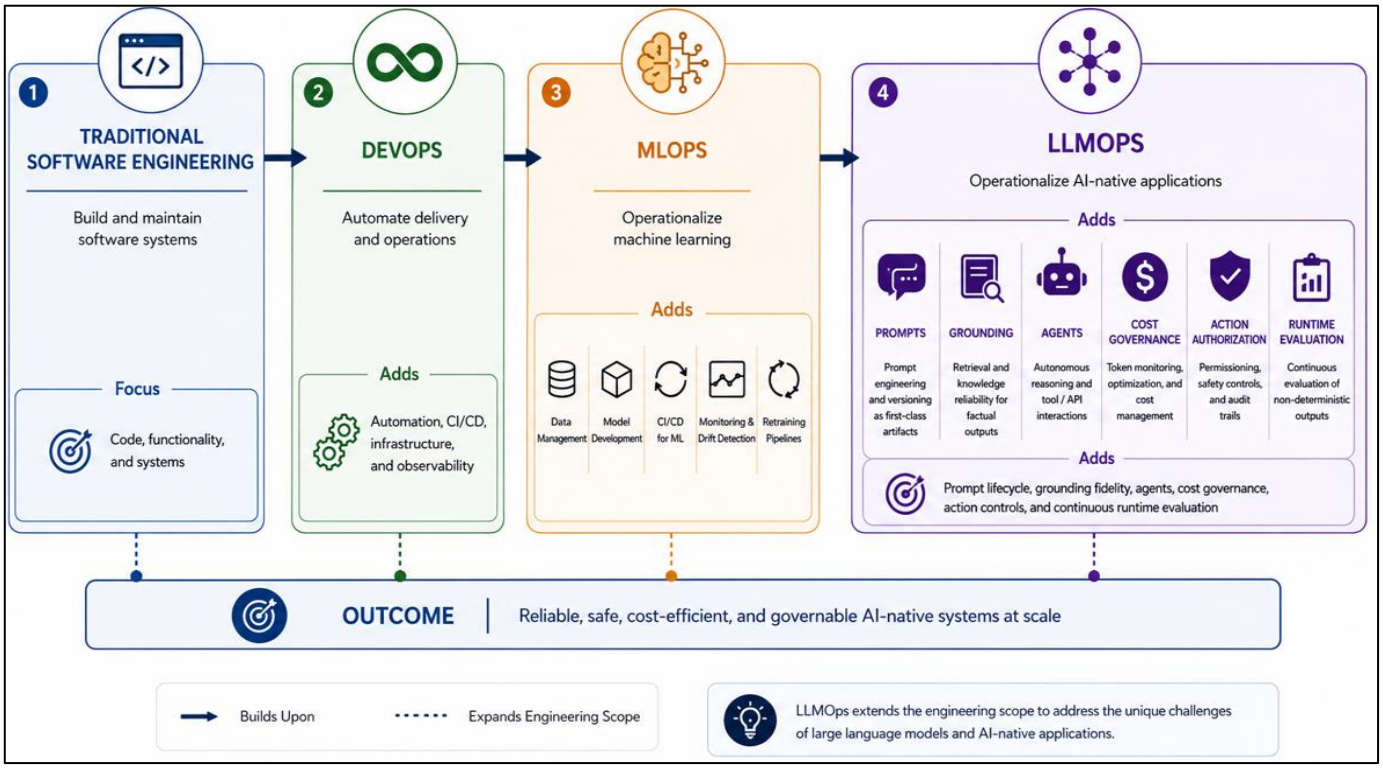

Fig 1 Evolution of AI Engineering Disciplines

Source: Developed by the author based on Kreuzberger et al. (2023), Skafidas and Tsoumakas (2023), White et al. (2023), and Wu et al. (2023)

Fig 1 shows that LLMOps is built on top of the existing paradigms of engineering, with an added layer of operations that is difficult to handle using traditional MLOps. LLMOps is not a replacement for MLOps, but rather an extension of the engineering scope that includes prompt lifecycle management, runtime reasoning, grounding mechanisms, autonomous agents, governance needs, and more relevant to AI-native applications.

➢ *Fundamental Differences between ML Systems and LLM Systems*

It's easy to see the differences between the concepts when all the things above are compared to the operational features of the traditional machine learning systems and the modern applications of the large language model. Table 1 lists the main engineering differences that support the view that LLMOps will be a new discipline and not just a new iteration of MLOps.

Table 1 Comparison of MLOps and LLMOps

Aspect	MLOps	LLMOps
Core artifact	Trained model	Prompt, orchestration logic, and foundation model
Data flow	Static training and inference pipeline	Dynamic context, retrieval, and runtime interaction
Prediction	Generally deterministic	Probabilistic and non-deterministic generation
Deployment	Organization-controlled models	External APIs and foundation model services
Evaluation	Accuracy, precision, recall, F1-score	Response quality, factuality, grounding, usefulness, and human evaluation
Monitoring	Infrastructure, latency, drift, resource utilization	Prompt performance, hallucinations, token usage, retrieval quality, runtime evaluation
Cost	Infrastructure and computational resources	Token consumption, API pricing, context length, inference routing
Risk	Prediction errors and model degradation	Hallucinations, unsafe agent actions, prompt injection, external dependency risks

Source: Developed by the author based on Kreuzberger et al. (2023), Paleyes et al. (2022), Kaddour et al. (2023), White et al. (2023), Bang et al. (2023).

Table 1 shows that LLMOps is more than just MLOps, it adds new engineering priorities. LLMOps is about prompts, dynamic retrievals, and out-of-control foundation models, while MLOps is about models and static data pipelines. Given that outputs are non-deterministic, the standard assessment from accuracy metrics is changing to evaluating the quality of reasoning, the factual reliability, and the ease of use to the user (Bang et al., 2023).

Similarly, monitoring now extends to real-time performance, grounding fidelity, real-time observability, and token usage, and operational costs are linked to API and token usage. These variations, which come with new governance and security issues such as hallucinations, prompt injection and autonomous agent actions, further emphasize the need for LLMOps to be established as its own software engineering practice.

III. THE FIVE ENGINEERING PRACTICES THAT DEFINE LLMOPS

LLMOps extends beyond traditional MLOps by addressing engineering challenges unique to AI-native applications. While the aim of MLOps is to deal with models, datasets and deployment pipelines, LLMOps has to address prompt-driven behavior, external foundation models, non-deterministic outputs, dynamic knowledge retrieval and autonomous agents. These variations necessitate new working methods, where LLMOps becomes a new engineering domain with a focus on prompt versioning, fidelity grounding, cost control, action authorization, and iterative testing.

➢ *Prompt Versioning as a First-Class Software Artifact*

For example, in LLM-based applications, prompts are now critical engineering artifacts that directly shape the systems' behavior. In contrast to conventional machine learning, the trained model itself is not always the central product, but rather its performance relies on prompt design. As a result, prompts need to be similarly version controlled, tested, documented, and rolled back as software source code (White et al., 2023).

Prompt versioning helps organizations monitor changes, performance, and safely deploy updates. Small variations in the prompts can make a huge difference in the answers you receive, making it all the more vital to keep an eye on your prompt lifecycle to ensure application quality, consistency, and compliance. Prompt engineering should be treated as a first-class citizen of LLMOps and follow a continuous lifecycle of prompt design, testing, deployment, monitoring and refinement, as shown in Fig 2.

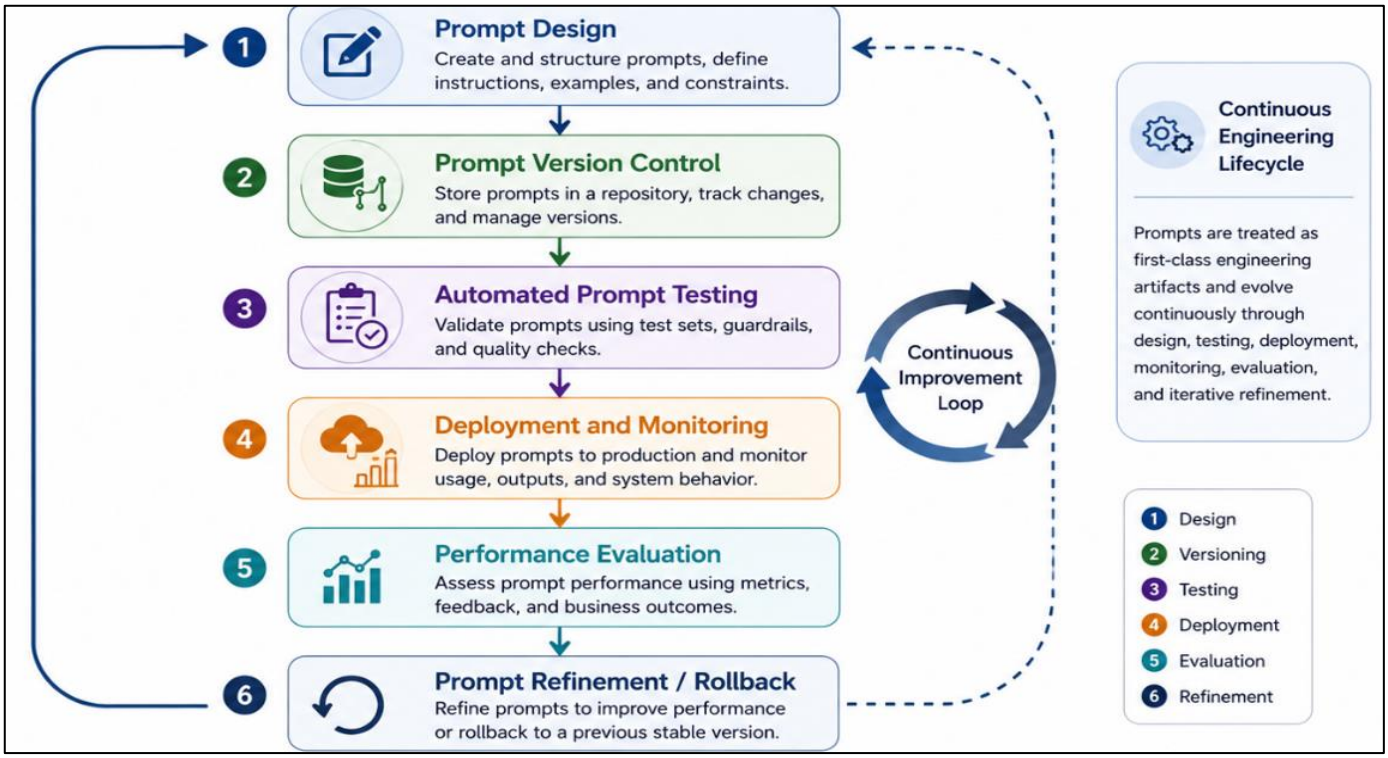

Fig 2 LLMOps Prompt Lifecycle Management Framework

Source: Developed by the author based on White et al. (2023) and Skafidas and Tsoumakas (2023).

Fig 2 illustrates that prompts must go through a continuous process of design, test, deployment, monitoring, and refinement. Prompt management is a key component of LLMOps, as the changes in prompts can drastically affect the system's behavior without altering the model itself.

➢ *Grounding Fidelity and Knowledge Reliability*

Grounding fidelity guarantees that LLM responses are accurate, reliable, and sourced from trusted information. Engineering techniques like Response Verification (RV), Source Validation, and Retrieval-Augmented Generation (RAG) are crucial as LLM can produce hallucinatory content. The ability to ground fidelities is a critical aspect of LLMOps, as highlighted by Kaddour et al. (2023) and Bang et al. (2023) who cite hallucinations and inconsistent reasoning as significant challenges for accurate deployment.

➢ *Cost Governance in Token-Based Computing*

Unlike traditional ML systems, LLM applications are charged based on token size as well as the size of their prompt and context length and the type of model used. Therefore, the cost governance part of LLMOps should include: Thus, cost governance in LLMOps should involve: Operational efficiency is currently a technical and economic issue for AI native systems, as highlighted by Skafidas and Tsoumakas (2023).

➢ *Action Authorization for Agentic AI Systems*

Agentic AI systems can run tools, APIs, and workflows on their own, putting new security and governance risks into play. For the safe operation of LLMops, authorization mechanisms, permission controls, audit trails and human oversight are required. As multi-agent systems become more capable, good governance becomes more crucial (Wu et al., 2023).

➢ *Evaluation under Non-Deterministic Outputs*

The output of LLM is inherently non-deterministic and cannot be captured by the traditional accuracy measures. Instead, LLMOps is based on an iterative process of testing with human feedback, quality metrics, and monitoring during runtime. The Continuous Evaluation CI/CD Pipeline emphasizes the need to move beyond model validation, to continuous operational assessment throughout deployment.

Table 2 Core Engineering Practices Distinguishing LLMOps from MLOps

LLMOps Practice	**Primary Engineering Objective**	**Operational Challenge Addressed**
Prompt Versioning	Manage prompts as software artifacts	Behavioral changes from prompt changes
Grounding Fidelity	Improve factual reliability	Hallucinations and unsupported responses
Cost Governance	Control token-based costs	Unpredictable operational expenses
Action Authorization	Secure autonomous execution	Unsafe agent behavior
Continuous Evaluation	Assess dynamic outputs	Non-deterministic responses

Source: Developed by the author based on White et al. (2023), Kaddour et al. (2023), Wu et al. (2023), Bang et al. (2023).

Table 2 highlights the key differences between LLMOps and MLOps. They tackle the issues of reliability, governance, cost and evaluation that are specific to AI-native applications, along with the growing recognition of LLMOps as a separate engineering discipline.

IV. CONSEQUENCES FOR SOFTWARE ENGINEERING RESEARCH AND PRACTICE

This rise of LLMOps has profound implications in software engineering research and practice. The rise in AI-supported applications is expanding the applications of AI into the business world, requiring software engineering communities to revisit their current development practices, assessment systems, and norms. New approaches are needed to move from MLOps to LLMOps, taking into account the distinct nature of generative AI systems.

➢ *Implications for Researchers*

LLMOps opens new research possibilities for software engineering researchers for AI lifecycle management, AI evaluation methods, reliability engineering, and human-AI collaboration. Current software engineering techniques were mainly developed for deterministic systems, while LLM-based applications demand techniques that can handle uncertainty and stochastic behavior.

There are many things that need to be explored in the future: automated prompt testing, LLM observability frameworks, evaluation benchmarks, and governance mechanisms, for instance. The presence of hidden operational failures in machine learning systems is pointed out by Tambon et al. (2022), indicating that such issues could be more complex in LLM applications because of the systems' dynamics. Researchers need to create methodologies that would enable them to detect failures on every prompt, retrieval system, model, and agent interaction. Fig 3 illustrates the overall research connections between the LLMOps components.

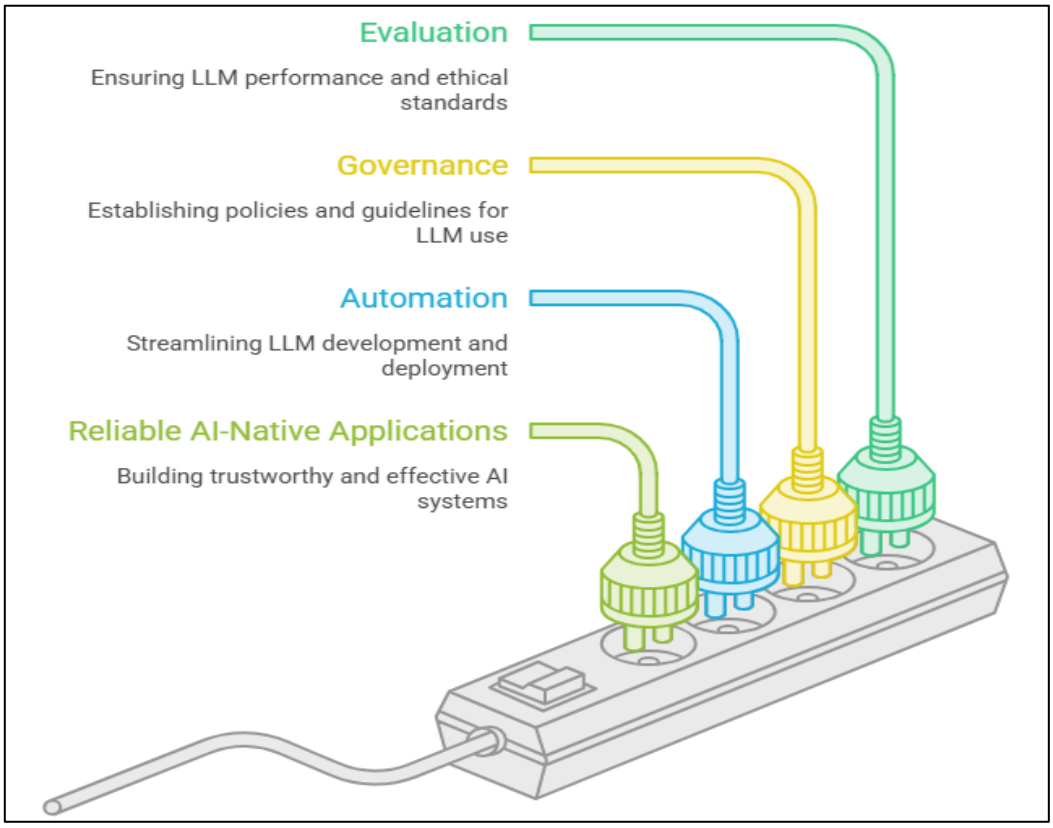

Fig 3 Research and Practice Ecosystem for LLMOps

Source: Developed by the author based on Kreuzberger et al. (2023), Kaddour et al. (2023), and Skafidas and Tsoumakas (2023).

As shown in Fig 3, LLMOps is a process that is not a matter of polishing up on one technical aspect; it is about collaboration between research areas. For AI-native applications to be effective, they must incorporate software engineering, AI evaluation, security, and governance views.

➢ *Implications for Industry*

LLMOps involves adapting to the organization for industry practitioners. Organizations transitioning to LLM applications need to create new engineering positions, processes, and governance. In addition to the traditional machine learning team might need, there may be a need for specialized professionals in prompt engineering, AI security, retrieval systems, and runtime evaluation.

While MLOps practices are still relevant, organizations need to enhance their operational capabilities to meet the specific needs of foundation model applications. As enterprises move towards LLMOps, the idea of matching technical tools to organizational goals as stated by Ruf et al. (2021) and by Mäkinen et al. (2021) is crucial.

➢ *Towards Standardization*

As the use of LLM applications continues to increase, there is a need for standardization. While LLMOps is still a nascent field of software engineering, there are no standardized frameworks, standards for evaluation, or governance models.

Table 3 shows that the adoption of standardization is crucial to make LLMOps a recognized engineering discipline. Shared frameworks will increase reliability, promote responsible adoption, and help organizations to handle more and more AI-native systems.

Table 3 Proposed Areas for LLMOps Standardization

Area	Standardization Need
Evaluation	Common quality and reliability benchmarks
Governance	AI safety and authorization frameworks
Monitoring	Runtime observability standards
Lifecycle Management	Prompt and model versioning practices

Source: Developed by the author based on Tabassam (2023), Vangala (2022).

Table 3 shows that the adoption of standardization is crucial to make LLMOps a recognized engineering discipline. Shared frameworks will increase reliability, promote responsible adoption, and help organizations to handle more and more AI-native systems.

V. RESEARCH AGENDA FOR THE LLMOPS COMMUNITY

The advent of LLMOps as an engineering discipline opens a wide research program beyond enhancing single language models, to building reliable, scalable, and governable AI-native software systems. Previous studies in MLOps have laid the groundwork for machine learning lifecycle management, but the operational nature of LLMs demands novel engineering strategies dealing with dynamic prompts, external model dependencies, and autonomous behavior, along with continuous runtime evaluation. Thus, systematic methodologies, tools and standards that can be used to support the next generation of AI applications should be the target of future LLMOps research.

A key research area is the timely design of life-cycles. As prompts become more and more an important software artifact that shapes the behaviour of the application, future research can focus on automated management techniques such as version control strategies, regression testing methods, optimization techniques, and impact analysis frameworks for prompt management. Further research is needed to systematically evaluate the necessity of change to arrive at changes made in a timely manner that ensures that system behaviour does not change unexpectedly and the application remains reliable (White et al., 2023).

Another key research challenge is the automation of the assessment of non-deterministic AI systems. Traditional software testing and machine learning validation techniques are not sufficient to validate systems with variable outputs. Continuous evaluation frameworks that will be able to measure factuality, reasoning quality, safety, usefulness, and alignment during operation in real world scenarios should be developed for future LLMOps research. A key direction is runtime observability approaches, which move the evaluation from one-off tests to continuous monitoring throughout the application lifecycle (The Continuous Evaluation CI/CD Pipeline: Shifting from Model Validation to Runtime Observability in LLMOps).

Further work is also required for secure, agent orchestration and authorization mechanisms. Autonomous actions in LLM applications are becoming more common, and researchers need to find ways to manage the autonomous actions, while also maintaining flexibility in the system. While multi-agent frameworks offer great promise for enhancing the capabilities of AI, they also present their own unique set of challenges, particularly in terms of accountability, access control, and operational safety (Wu et al., 2023).

Also, optimizing the economy and managing resources are important research issues. Unlike the traditional machine learning systems, LLM applications come with a cost that is dependent on the number of tokens, the type of model selected, and the complexity of context in which it is used. Going forward, cost-aware architectures, intelligent model routing, caching strategies, and resource optimization techniques, which strike a balance between performance and affordability, could be explored further (Skafidas & Tsoumakas, 2023).

Lastly, there is a need for LLMOps community to focus on establishing some standard engineering frameworks and governance models. Just as MLOps has standardized practices for deploying machine learning models, LLMOps demands standardized methods for evaluating, monitoring, securing, and managing the lifecycle of LLMs.Like MLOps which paved the way for common practices in deploying machine learning models, LLMOps calls for a common approach to evaluating, monitoring, securing, and managing LLM lifecycles. The development of these foundations will facilitate the transition of research and practice from experimental adoption to engineering practice.

In general, the future of LLMOps research is focused on developing engineering principles that address the operational nuances of AI-native apps. These challenges, if tackled systematically, can pave the way for LLMOps to become a well-developed discipline, enabling the creation of reliable, secure, and scalable intelligent systems.

VI. CONCLUSION

The advent of large language models (LLMs) has revolutionized the way AI systems are supposed to operate, which poses challenges for traditional MLOps approaches. In this paper, the authors have proposed that LLMOps should be considered as a separate engineering discipline, not just an extension of MLOps. Although MLOps successfully defined best practices in data management, model deployment, monitoring, and retraining (Kreuzberger et al., 2023; Ruf et al., 2021), AI-native applications are facing new engineering challenges that involve prompt engineering, grounding reliability, token-based cost governance, and the assessment of non-deterministic outputs.

The paper introduces five key engineering practices prompt versioning, fidelity to the ground truth, cost control, action authorization, and continuous evaluation to help conceptualize the distinct operational life cycle of systems based on the LLM. These practices reveal the critical need for coordinated management of prompts, retrieval, external foundation models, runtime observability, and human oversight to ensure the reliability of AI-native applications. This view aligns with previous studies that also indicate that operational failures are typically not due to the model itself but more often reflect failures of the system (Paleyes et al., 2022; Tambon et al., 2022), as well as recent LLM research that continues to discuss issues of reliability, hallucination, reasoning quality, and responsible deployment (Kaddour et al., 2023; Bang et al., 2023).

It is important for the software engineering community to acknowledge LLMOps as a distinct field of study with its own methods, standards, and governance structures to formally recognize the impact of its emergence and advancement in the field of software development. The research area talked about in Section 5 sets the groundwork for the future progress of this new field and of future cooperation between researchers, practitioners and industry. In order to create reliable, secure, scalable and trusted AI-native applications, robust LLMOps practices will be crucial.

REFERENCES

[1]. Kreuzberger, D., Kühl, N., & Hirschl, S. (2023). Machine learning operations (MLOps): Overview, definition, and architecture. *IEEE Access*, *11*, 3366–3379. https://doi.org/10.1109/ACCESS.2023.3262138

[2]. Tambon, F., Laberge, G., An, L., Khomh, F., & Antoniol, G. (2022). Silent bugs in machine learning systems: An empirical study. *IEEE Transactions on Software Engineering*, *48*(12), 4810–4825. https://doi.org/10.48550/arXiv.2112.13314

[3]. Paleyes, A., Urma, R. G., & Lawrence, N. D. (2022). Challenges in deploying machine learning: A survey of case studies. *ACM Computing Surveys*, *55*(6), 1–29. https://doi.org/10.1145/3533378

[4]. Mäkinen, S., Skogström, H., Laaksonen, E., & Mikkonen, T. (2021, May). Who needs MLOps: What data scientists seek to accomplish and how can MLOps help?. In *2021 IEEE/ACM 1st workshop on AI engineering-software engineering for AI (WAIN)* (pp. 109-112). IEEE. https://doi.org/10.48550/arXiv.2103.08942

[5]. Dhanekula, A., & Miah, M. R. (2022). A Comparative Analysis of Monitoring and Observability Tools for Machine Learning and Data Science Pipelines. *American Journal of Interdisciplinary Studies*, *3*(03), 99-134. https://ajisresearch.com/index.php/ajis/article/view/122

[6]. Kaddour, J., Harris, J., Mozes, M., Bradley, H., Raileanu, R., & McHardy, R. (2023). Challenges and applications of large language models. *arXiv preprint arXiv:2307.10169. https://doi.org/10.48550/arXiv.2307.10169*

[7]. Tabassam, A. I. (2023). MLOps: A step forward to enterprise machine learning. *arXiv preprint arXiv:2305.19298*. https://doi.org/10.48550/arXiv.2305.19298

[8]. Wu, Q., Bansal, G., Zhang, J., Wu, Y., Zhang, S., Zhu, E., Li, B., Nguyen-Tuong, K., & Wang, C. (2023). AutoGen: Enabling next-gen LLM applications via multi-agent conversation framework. *arXiv preprint arXiv:2308.08155*. https://doi.org/10.48550/arXiv.2308.08155

[9]. Skafidas, N., & Tsoumakas, G. (2023). From model-centric to data-centric and pipeline-centric AI: Operationalizing the deployment of large language models. *Information*, *14*(9), 492. https://doi.org/10.3390/info14090492

[10]. White, J., Fu, Q., Hays, S., Sandborn, M., Olea, C., Gilbert, H., Elnashar, A., Spencer-Smith, J., & Schmidt, D. C. (2023). A prompt pattern catalog to enhance prompt engineering with ChatGPT. *arXiv preprint arXiv:2302.11382*. https://doi.org/10.48550/arXiv.2302.11382

[11]. Bang, Y., Cahyawijaya, S., Lee, N., Dai, W., Su, J., Wilie, B., Choi, H., Ziems, C., Yu, J., & Fung, P. (2023). A multitask, multilingual, multimodal evaluation of ChatGPT on reasoning, hallucination, and objectivity. *arXiv preprint arXiv:2302.04023*. https://doi.org/10.48550/arXiv.2302.04023

[12]. Ruf, P., Madan, M., Reich, C., & Ould-Abdeslam, D. (2021). Demystifying MLOps and Presenting a Recipe for the Selection of Open-Source Tools. *Applied Sciences*, *11*(19), 8861. https://doi.org/10.3390/app11198861

[13]. The Continuous Evaluation CI/CD Pipeline: Shifting from Model Validation to Runtime Observability in LLMOps. https://www.yuktabpublisher.com/index.php/TCT/article/view/152

[14]. Vangala, V. (2022). MLOps in Practice: A Framework for Scalable AI Model Deployment, Monitoring, and Retraining. *International Journal of Machine Learning Research in Cybersecurity and Artificial Intelligence*, *13*. https://zenodo.org/records/15570286

[15]. Theodoropoulos, T., Makris, A., Boudi, A., Taleb, T., Herzog, U., Rosa, L., ... & Dazzi, P. (2022). Cloud-based xr services: A survey on relevant challenges and enabling technologies. *Journal of Networking and Network Applications*, *2*(1), 1-22. https://dx.doi.org/10.33969/J-NaNA.2022.020101