Industry-grade MLOps: an Architecture Proposal

Abstract

The exploitation of Artificial Intelligence (AI) models shall always be limited by their efficiency and effective fruition, from data provenance, traceability, quality to deployability and evolvability of data pipelines around such AI, collectively amounting to the Machine-Learning operations (MLOps) discipline. Through industrial action design research, this paper offers a preliminary view into the requirements and design challenges posed by the aforementioned discipline, starting from the functional and non-functional problems in our position, namely, a scale-up Small-Medium Enterprise (SME) interested in providing an MLOps platform with commercial and open-source components; our position draws from experiences in developing said solution, with a focus on its data provenance, traceability and quality-by-design. Our early prototype and its preliminary implementation show promise in delivering efficient and effective MLOps but our (limited) prototype scope calls for considerable efforts and validation before its production-readiness. We document and discuss the challenges and design principles learned while driving the inception of the prototype, its current status as well as the strengths and limitations of the proposed approach.

Keywords

MLOps, Software Engineering for AI, Software Operations

ACM Reference Format:

. 2018. Industry-grade MLOps: an Architecture Proposal. In *Companion Proceedings of the 33rd ACM Symposium on the Foundations of Software Engineering (FSE '25), June 23–27, 2025, Trondheim, Norway*. ACM, New York, NY, USA, 2 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Artificial Intelligence (AI) has emerged as a powerful tool capable of processing vast amounts of data and identifying complex patterns that might be imperceptible to traditional analytical methods. However, the practical application of AI in complex domains—e.g., climate change or industry-scale defect prediction—is not without its challenges [7]. The effectiveness of AI models depends not only on the sophistication of the algorithms but also on the efficiency and robustness of the data pipelines—or MLOps—that forage and deliver these models. Issues such as data provenance, traceability, quality, deployability, and the ability to evolve MLOps are critical non-functional requirements that must be addressed to maximize the utility and value of delivering AI. Our approach—by means of a design-scientific action research approach [6]—leverages on a MLOps prototype to ensure that the critical aspects of data provenance, traceability, and quality are embedded by design, thus enhancing the overall reliability and efficiency of delivering AI-driven software. The prototype offered in this paper is an early birds'eye view over the consolidated architecture we are targeting. The end solution reflects a concrete MLOps platform prototype—called `chaM3Leon`[1]—for further research and use by practitioners and researchers alike.

2 Proposed Architecture

The platform's architecture is a tailoring of the well-known lambda architecture pattern [9] organized into seven principal layers (see also Fig. 1):

- Batch Layer: Built on Spark Structured Streaming, this component is dedicated to processing non-real-time data streams. Specifically, streams originating from one or more Kafka[2] topics undergo a customizable preprocessing, with the processed results saved on the Hadoop[3] Distributed File System (HDFS). In parallel, a process generates partial statistics stored on a non-relational database, Cassandra[4], further enhancing data accessibility for historical analyses [3].
- Speed Layer: This layer manages real-time data streams, calculating partial statistics on the latest incoming data to provide rapid, though preliminary, insights. Like the Batch Layer, the Speed Layer is based on Spark Structured Streaming. Unlike the Batch Layer, it does not store data on HDFS, offering instead immediate data access for applications requiring real-time updates—a critical capability for continuous monitoring scenarios [1].
- Harvesting Layer (Harvester): The Harvester is responsible for gathering data from heterogeneous external sources in cases where the Batch and Speed Layers are insufficient, such as collecting historical data from external archives or managing data formats unsuitable for the Batch Layer. Its primary goal aligns with that of the Batch Layer: to process and shape data for optimal consumption by machine learning algorithms [8]. The prototype supports widely used data formats in geoclimatic analysis, initially GeoJSON and TIF/TIFF, and its ability to integrate with external APIs and data services enhances its applicability across diverse use cases.
- ML Layer: This layer handles the modeling phase, reading data from HDFS that the Batch Layer and/or Harvester provide and using it as the basis for defining a machine learning pipeline. This pipeline produces a model stored on HDFS, while inference results are saved in Cassandra, thereby ensuring that insights are readily accessible. Integrating machine learning into environmental data architectures has been shown to improve predictive accuracy and support

ESEC/FSE 2025, Trondheim, NW

ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

[1] https://github.com/Smart-Shaped/chaM3Leon/tree/public
[2] https://kafka.apache.org/
[3] https://hadoop.apache.org/
[4] https://cassandra.apache.org/_/index.html

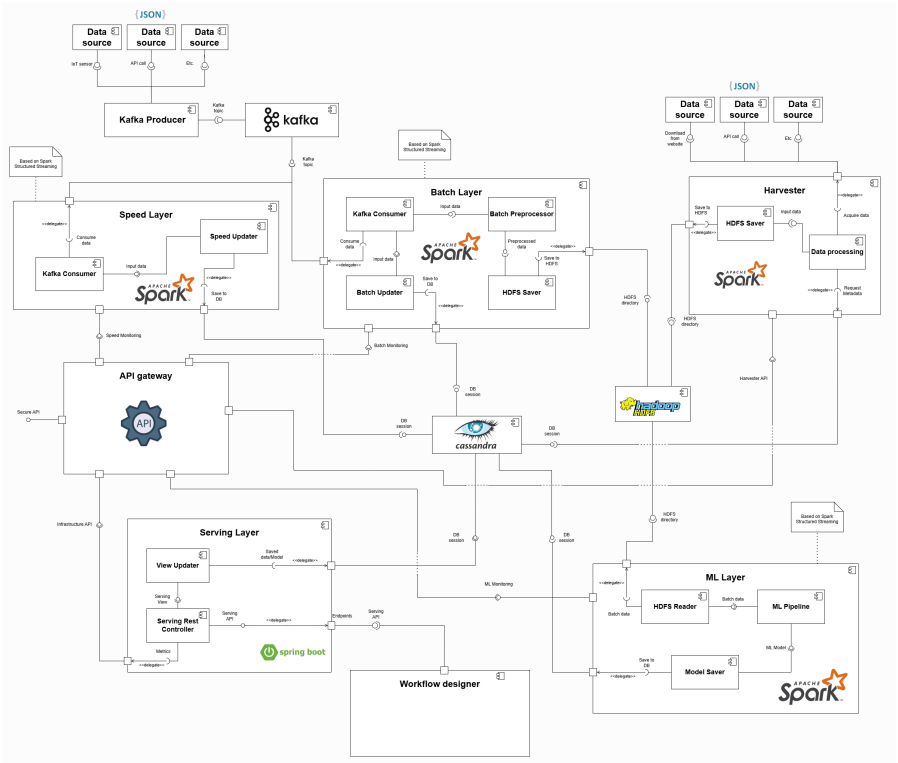

Figure 1: an architectural diagram—elaborated following the guidelines for structuring UML-based component/connector views of software architectures [4]—of our prototype.

near-real-time insights, essential in data-driven climate science [5].

- Serving Layer: The Serving Layer aggregates data from Cassandra, including summary statistics and machine learning inferences. This layer's purpose is to facilitate access to processing results through exposed APIs. Providing accessible outputs through APIs is crucial for integrating AI predictions into decision-making processes, especially in climate science, where users need straightforward access to data for responsive action [2].
- Workflow Designer: An independent component, integrated within the Serving Layer, designed to enable no-code creation of custom pipelines. Through exposed endpoints, the Serving Layer collects user-defined inputs and translates them into operational configurations for the platform. This functionality facilitates the generation of tailored pipelines regarding layer utilization, selection of machine learning algorithms, and integration of diverse data sources. This component remains as a baseline (in Fig. 3) in the prototyping stage, as the initial development phase prioritized the platform's core functional capabilities.
- API Gateway: Responsible for managing communication with the architecture through APIs, the API Gateway is designed to collect metrics from the various layers in operation. Additionally, it offers the capability to trigger workflows generated by the Workflow Designer. These functionalities are accessible via two types of exposed APIs: "secure APIs," accessible externally and primarily used for retrieving inference results; and "Infrastructure APIs," accessible from the Serving Layer, which are employed to supply metrics and define workflows.

References

[1] Chen, A., and Torres, M. Real-time data ingestion techniques for environmental monitoring. *Journal of Real-Time Data Systems 17*, 2 (2023), 102–115.

[2] Jain, K., and Li, H. Api design for ai-driven environmental data applications. *International Journal of Environmental Informatics 25*, 1 (2024), 66–81.

[3] Kumar, R., and Zhao, X. Streamlining batch data pipelines for environmental data applications. *Journal of Cloud Computing 12*, 3 (2023), 193–209.

[4] Medvidovic, N., Rosenblum, D. S., Redmiles, D. F., and Robbins, J. E. Modeling software architectures in the unified modeling language. *ACM Trans. Softw. Eng. Methodol. 11*, 1 (January 2002), 2–57.

[5] Patel, H., and Lin, S. Integrating ml pipelines for predictive modeling in climate data architectures. *Geoscientific Model Development Discussions 17*, 1 (2024).

[6] Sein, M. K., Henfridsson, O., Purao, S., Rossi, M., and Lindgren, R. Action design research. *MIS Quarterly 35*, 1 (2011), 37–56.

[7] Tamburri, D. A., van Mierlo, V. R., and van den Heuvel, W.-J. Big data for the social good: The drought early-warning experience report. *IEEE Transactions on Big Data 9*, 3 (2023), 773–791.

[8] Wang, D., and Singh, V. Integrating external data for climate modeling: A review of best practices. *Sustainable Data Science 4* (2023).

[9] Yamada, T., and Choi, Y. Efficient management of streaming and batch climate data using lambda architecture. *Journal of Big Data Research 29* (2022).