An Efficient Microservices Architecture for MLOps

Seol Roh
Department of Computer Science and Engineering
Kyung Hee University
Gyeonggi-do 17104, Republic of Korea
seven800@khu.ac.kr

Ki-Moon Jeong
Center for Supercomputing Technology Development
Korea Institute of Science and Technology Information
Daejeon-si 34141, Republic of Korea
kmjeong@kisti.re.kr

Hye-Young Cho
Center for Supercomputing Technology Development
Korea Institute of Science and Technology Information
Daejeon-si 34141, Republic of Korea
chohy@kisti.re.kr

Eui-Nam Huh
Department of Computer Science and Engineering
Kyung Hee University
Gyeonggi-do 17104, Republic of Korea
johnhuh@khu.ac.kr

***Abstract*— In a microservices architecture, each service has a database. Hence, it is important to communicate and synchronize data between services. The SAGA pattern is a traditional microservices architecture pattern, and the command query responsibility segregation (CQRS) pattern has recently attracted increasing attention. Machine learning model operation management (MLOps) aims to stably deploy and maintain the system by preprocessing big data and learning machine learning models. Data processing in the microservices architecture is important because considerable data is used. This paper proposes an appropriate architecture for each microservice to perform efficiently in the MLOps environment.**

Keywords— microservice; Distributed Cloud; SAGA; CQRS; MLOps

I. INTRODUCTION

Society has become big data-centered in the big data generation of the 4th industrial revolution. Fields, such as artificial intelligence and machine learning, are in the limelight [1]. Accordingly, interest in machine learning model operation management (MLOps) and microservices architecture (MSA) has also increased. MLOps [2] is used when the DevOps (development and operations) [3] process is applied to a machine learning system. In other words, based on the huge amount of data, the data is preprocessed and verified inside the ML process; a learning method is determined, and a model is developed to perform large-scale learning and distribution. In this paper, ML Pipeline services, such as data preprocessing, verification, model learning, and model deployment required for MLOps configurations, are divided according to the MSA.

Ruibo Chen presented an AIOps system based on microservice platforms but did not deal directly with MLOps [4]. Tim Raffin presented a design for the MLOps structure and deployment process as a reference architecture for operationalizing a machine learning model [5]. Ruibo Chen and Tim Raffin performed MSA designs that did not consider data transactions. To best of our knowledge, no research has been conducted that has contributed to improving MLOps performance by combining MLOps and MSA. Therefore, when designing an MSA in artificial intelligence and machine learning using big data, it is necessary to provide services to users quickly by considering data flow and transactions.

This study focused on MSA to obtain better processing performance than the existing monolithic architecture using an MLOps system [6], [7]. This paper proposes a suitable architecture using representative patterns, the SAGA pattern [8], and command query responsibility segregation (CQRS) pattern [9].

II. RELATED WORK

A. Microservices Architecture

Unlike the monolithic structure in which the service structure is integrated into one, the microservice structure is composed of several microservice units [6], [7]. The advantage is that microservices can be deployed independently and expanded, and there are no restrictions on programming languages between microservices [7]. Each service runs while communicating with lightweight mechanisms, such as RESTful APIs [7], [9]. Elements, such as those for each service to communicate, a registry for each service, service monitoring and management, migration tools, and deployment process tools for smooth deployment of services, are sometimes necessary to apply the microservice structure to the system smoothly. Recent research has indicated that Microservices have been applied successfully by Netflix and SoundCloud in their cloud computing applications [10], [11].

B. SAGA Pattern

The SAGA design pattern [8] maintains data consistency across microservices in distributed transaction scenarios. In the existing monolithic architecture, data consistency is maintained using the transaction function of the DBMS itself. In MSA, DB exists for each service. The SAGA pattern maintains data consistency as services and DBs are distributed [12]. When a transaction fails while processing an event between services in the MSA environment, data consistency and atomicity are guaranteed by providing a failure compensation transaction to services whose work has been completed [8], [13]. The SAGA pattern has been used elsewhere. In [14], the authors proposed

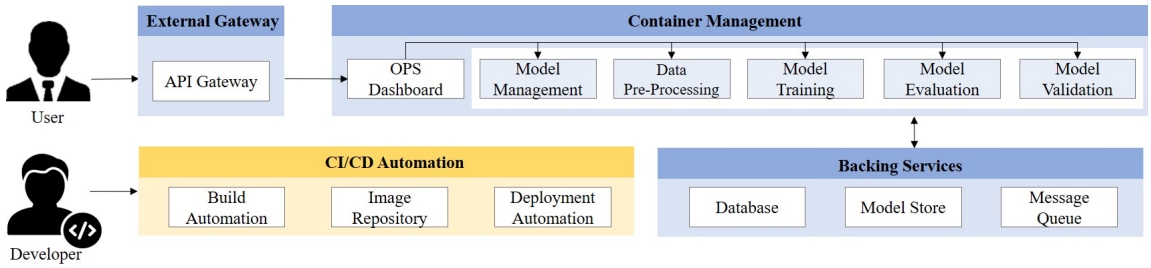

Figure 1. Microservices Architecture Components for MLOps

an enhanced saga pattern for distributed transactions in MSA. This saga pattern has atomicity, consistency, and durability, except for isolation. The authors in [14] proposed a SAGA pattern of ACID(Atomicity, Consistency, Isolation, Durability), that solved the lack of read isolation.

C. CQRS Pattern

In general, insert, update, delete, and select data are all performed in the same storage as a request from business logic [16]. They are divided into commands that change the system state and those that query the system state. The most frequently used request is the part of querying the system state. If the insert, update, delete, and select functions are all included in the service, there is a disadvantage that all functions must be expanded according to the select request frequency. A CQRS [16], [17] pattern that separates reading and writing has been used to solve this problem. This pattern can prevent resource deadlock when MSA scales out and runs in service units.

D. MLOps

MLOps realizes at least five functions together, including data collection, data transformation, ML learning, ML deployment, and user service [18]. It is a system that performs a series of MLOps pipelines to enable users to learn seamlessly with machine learning. Sasu Mäkinen et al. studied the importance of MLOps in the context of data scientists' daily activities from 331 professionals in the ML domain [19]. According to Sasu Mäkinen's research, the most important challenge in ML is data, and data scarcity and data accessibility are important tasks. Interest in infrastructure issues in the MLOps environment is growing.

III. SYSTEM ARCHITECTURE

A. Problem Scenario

Compared to the MLOps structure in MSA, the MLOps structure in the monolithic architecture has a limited problem in processing speed compared to the MSA, which is the Asynchronous Non-Blocking method, due to the Synchronous Blocking method. Therefore, the user might have to wait until the processing is completed using the MLOps system. In addition, there is a limited problem with the data read/write processing speed. Fig 2 presents the monolithic architecture. Because OPS dashboard, ML Pipeline services, and Data Access Layer are all connected to one database, they do not operate independently. As a result, it is very difficult to maintain a smooth MLOps service in the monolithic architecture as the number of users increases.

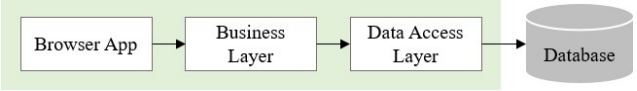

Figure 2. Monolithic Architecture

B. Need Microservices Architecture for MLOps

Fig 1 shows the proposed designed MSA. The users access Operation dashboard through API Gateway. In Ops dashboard, the users can manage model history through Model Management Service and register data to be trained. Data can be pre-processing, evaluation, and the result of validation can be checked. All these services are performed independently, and there is no need to wait for each service to be complete. Fig 3 shows the MLOps Model Life Cycle. Services within each pipeline are also divided into microservice units and configured in an MSA structure to ensure independence between services.

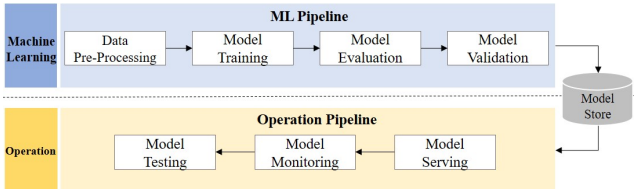

Figure 3. Model Life Cycle for MLOps

In the MSA structure, the CI/CD of each microservice can be automated, and Build Automation and Deployment Automation can be performed through the Image Repository. In reality, where various models are continuously being developed, the MSA is suitable for the MLOps environment because of the nature of the machine learning field.

The backing service refers to all services available through the network when the service is running. Examples include databases, message queues, and Simple Mail Transfer Protocol(SMTP). A message queue method is essential because MSA is an asynchronous, non-blocking method. It is common for each service to have a database. To this end, an asynchronous communication pattern using a message queue must be used to ensure consistency with the data of other services.

C. Proposed Method

Fig 4 presents the proposed efficient microservices architecture in MLOps environments. By applying the database-per-service pattern, each service must have a database. Loose coupling is a key feature of MSA because

each microservice can store and retrieve data independently from its database.

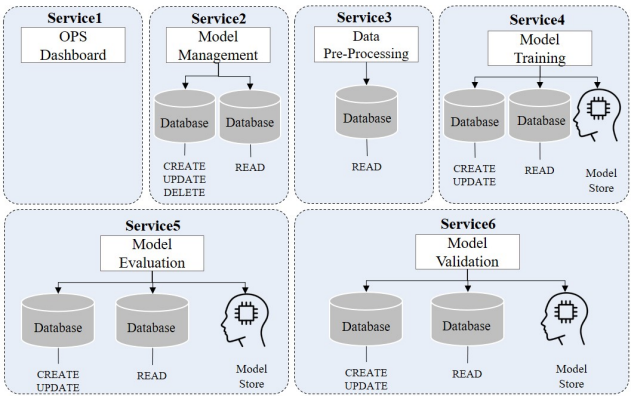

Figure 4. Microservices Architecture for MLOps

Among the microservices in Fig. 4, the Model Management Service is the service that calls the database most frequently. The Create, Update, Delete Database, and Read Database are separated by applying the CQRS pattern because it is connected to the user terminal through the Operation dashboard service. The Model Training Service and Inference Service are equally applied, and Model Store is used because it trains and infers models. Accordingly, the orchestration-based SAGA Pattern was used to handle complex transactions spanning multiple microservices.

IV. CONCLUSION

This paper proposed efficient MSA recommendations in MLOps environments. Using MSA's patterns, the databases of each MLOps service do not load when processing large amounts of data, and users can use machine learning services quickly.

Future research will reflect the core-edge structure to cache MLOps services from the core cloud to the edge cloud so that the service can be provided to the user in real-time based on the situation. In addition, the infrastructure management of AIOps services can be managed efficiently, and a flexible system that can quickly process big data will be studied through research that recommends an optimized microservice structure tailored to AIOps.

ACKNOWLEDGMENT

This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.RS-2023-00220631, Edge Cloud Reference Architecture Standardization for Low Latency and Lightweight Cloud Service) and the Korea Institute of Science and Technology Information(KISTI)(Research of MLOps Key Technology for Cloud Based HPC Task Processing).

REFERENCES

[1] SOROOSHIAN, Shahryar, and Shrikant PANIGRAHI, "Impacts of the 4th Industrial Revolution on Industries," Walailak Journal of Science and Technology (WJST) vol. 17, no. 8, pp. 903-915, August 2020.

[2] Sridhar Alla and Suman Kalyan Adari, What Is MLOps?. In: Beginning MLOps with MLFlow. Apress, CA:Berkeley, 2021.

[3] C. Ebert, G. Gallardo, J. Hernantes, and N. Serrano, "DevOps," IEEE Software, vol. 33, no. 3, pp. 94-100. May-June 2016.

[4] Chen, Ruibo, and Wenjun Wu. "Parallelizing Automatic Model Management System for AIOps on Microservice Platforms," Euro-Par 2021: Parallel Processing Workshops: Euro-Par 2021 International Workshops, Lisbon, Portugal, August 30-31, 2021, Revised Selected Papers, Cham: Springer International Publishing, 2022. p. 376-387.

[5] T. Raffin, T. Reichenstein, J. Werner, A. Kühl, and J. Franke, "A reference architecture for the operationalization of machine learning models in manufacturing," Procedia CIRP, vol. 115, pp. 130-135, 2022.

[6] Johannes Thönes, "Microservices," IEEE software, vol. 32, no. 1, pp. 116-116, January-February 2015.

[7] Xabier Larrucea, Izaskun Santamaria, Ricardo Colomo-Palacios, and Christof Ebert "Microservices," IEEE Software, vol. 35, no. 3, pp. 96-100, May-June 2018.

[8] R. H. Campbell, and P. G. Richards, "SAGA: A system to automate the management of software production," national computer conference, pp. 231-234, May 1981.

[9] Singh, Vindeep, and Sateesh K. Peddoju, "Container-based microservice architecture for cloud applications," 2017 International Conference on Computing, Communication and Automation (ICCCA), pp. 847-852, May 2017.

[10] Sudhir Tonse, Microservices at netflix - challenges of scale, August 2014, [online] Available: http://www.lideshare.net/stonse/microservices-at-netflix.

[11] Phil Calçado, Building products at soundcloud-part iii: Microservices in scala and finagle, June 2014, [online] Available: https://developers.soundcloud.com/blog/building-products-at-soundcloud-part-3-microservices-in-scala-and-finagle.

[12] H. Garcia-Molina, and K. Salem, "Sagas," ACM Sigmod Record, vol. 16, no. 3, pp. 249-259, December 1987.

[13] Aydin, Sahin, and Cem Berke Çebi, "Comparison of Choreography vs Orchestration Based Saga Patterns in Microservices." 2022 International Conference on Electrical, Computer and Energy Technologies (ICECET), pp. 1-6, July 2022.

[14] Daraghmi, Eman, Cheng-Pu Zhang, and Shyan-Ming Yuan, "Enhancing Saga Pattern for Distributed Transactions within a Microservices Architecture," Applied Sciences, vol. 12, no. 12, June 2022.

[15] Zhong, Yifan, Wei Li, and Jing Wang, "Using event sourcing and CQRS to build a high performance point trading system," 2019 5th International Conference on E-Business and Applications(ICEBA), pp. 16-19, February 2019.

[16] D. Betts, J. Dominguez, G. Melnik, F. Simonazzi, and M. Subramanian, Exploring CQRS and Event Sourcing: A journey into high scalability, availability, and maintainability with Windows Azure, 2013.

[17] Kabbedijk, Jaap, Slinger Jansen, and Sjaak Brinkkemper, "A case study of the variability consequences of the CQRS pattern in online business software," 17th European Conference on Pattern Languages of Programs, pp 1-10, July 2012.

[18] Tamburri, and Damian A, "Sustainable mlops: Trends and challenges," 2020 22nd international symposium on symbolic and numeric algorithms for scientific computing (SYNASC), pp 17-23, September 2020.

[19] Mäkinen, H. Skogström, E. Laaksonen, and T. Mikkonen, "Who needs MLOps: What data scientists seek to accomplish and how can MLOps help?," 2021 IEEE/ACM 1st Workshop on AI Engineering-Software Engineering for AI (WAIN), pp. 109-112, May 2021.