Secure Collaborative Model Training with Dynamic Federated Learning in Multi-Domain Environments

Anestis Dalgkitsis, Alexandros Koufakis, Jorrit Stutterheim, Aleandro Mifsud, Priyanka Atwani,
Leon Gommans, Cees de Laat, Chrysa Papagianni, and Ana Oprescu
Informatics Institute
University of Amsterdam
Amsterdam, The Netherlands
{a.dalgkitsis, a.m.koufakis, a.mifsud, l.gommans, c.t.a.m.delaat, c.papagianni, a.m.oprescu}@uva.nl,
priyankaatwani@live.com, and jorrit.stutterheim@gmail.com

***Abstract*—According to the European Union Aviation Safety Agency (EASA), AI-based algorithms, combined with extensive fleet data, could enable early detection of potential engine failures, leading to proactive predictive maintenance in air travel. At a global level, the Independent Data Consortium for Aviation (IDCA) recognizes the potential of collaborative data sharing in the airline industry. However, data ownership-related issues, such as privacy, intellectual property, and regulatory compliance, pose significant obstacles to realizing the vision of combining fleet data to improve predictive maintenance algorithms. In this paper, we use NASA's Turbofan Jet Engine Dataset (N-CMAPSS) to demonstrate how airlines could leverage the power of Federated Learning (FL) and microservices, to collaboratively train a global Machine-Learning (ML) model that can enable airline companies to utilize their data for predictive maintenance, while maintaining control.**

***Index Terms*—Microservices, Data Privacy, Federated Learning, Data Aggregation, Multi-Domain Networking.**

I. Introduction

The European Union Aviation Safety Agency (EASA) emphasizes that the integration of AI-driven predictive maintenance with the extensive fleet data now available has the potential to significantly enhance the early detection of possible engine failures [1]. This approach enables proactive maintenance, which could play a crucial role in improving safety and preventing incidents in the future of air travel. On a global scale, the Independent Data Consortium for Aviation (IDCA) highlights the impact of collaborative data sharing within the airline industry, recognizing its potential to drive significant advancements in aviation safety and efficiency [2].

Federated Learning (FL) has emerged as a key technique in collaborative data training, enabling multiple organizations to train machine learning models on decentralized data, while maintaining data privacy and sovereignty. This method offers enhanced privacy and security compared to traditional approaches, making it particularly valuable in sectors where data sensitivity is critical, such as healthcare, finance, and aviation [3].

Microservices architecture is expanding the possibilities for data-sharing methods and marketplaces by decomposing applications into loosely coupled, independently deployable, and verifiable services. This approach enables seamless data exchange and integration across various platforms, with the potential to drive innovation and increase efficiency in data utilization [4].

Furthermore, advancements in next-generation of networking technology are leading to the development of a programmable internet architecture that supports network slicing, virtualization at a deeper level and programmable networking. The combination of these technologies enable the creation of multiple virtual networks on physical infrastructures, each tailored to specific service and application needs. Network slicing is essential for the future of internet services, ensuring optimal performance, security, and flexibility.

However, data ownership challenges, including privacy concerns, intellectual property rights, and regulatory compliance, present substantial obstacles to the realization of collaborative model training in a shared infrastructure, particularly among international airlines [1]. These challenges present significant obstacles to achieving the vision of collaborative model training among key industries, particularly international airlines, which are the focus of our work. Without such a framework, the potential benefits of collaborative data training cannot be fully realized.

The goal of this paper is to demonstrate how FL could be deployed using microservices in a multi-tenant and multi-domain network by utilizing FABRIC, a transatlantic network slice, described in Section II-C, and DYNAMOS, analyzed in depth in Section II-B, a tool that enables dynamic microservices deployment based on set policies. DYNAMOS allows each party to maintain control of their data, privacy concerns are taken into account by design, and contractual agreements are enforced.

We validate this approach through a use case relating to the EASA, detailed in Section III, using DYNAMOS as middleware to experiment with data-sharing scenarios. This is deployed on FABRIC to prove that international airline collaborative critical data exchange through FL training is possible with the current state of the art in an actual distributed environment. The key contributions can be summarized as follows:

- We demonstrate FL on an architecture that adheres to the standards of the airline industry, and enables support for collaborative model training for predictive maintenance algorithms.

- We show how this architecture, based on microservices, allows digital data exchange while ensuring privacy and contractual agreements by design.
- Finally, we evaluate this proposed architecture by deploying it on the FABRIC network slices, and orchestrating the services with the DYNAMOS middleware.

II. Background

In this section, we introduce the core concepts and tools used in this work, FL, DYNAMOS and the FABRIC testbed.

A. Federated Learning

FL is a decentralized approach to training machine learning models, initially introduced by Google [3]. Unlike traditional centralized training, which aggregates data into a central repository, FL allows multiple clients to collaboratively train a shared, global model while keeping individual data sources local. This approach minimizes data transfer, enhances privacy, and leverages diverse, heterogeneous datasets distributed across different environments. Figure 3 provides a schematic representation of a multinode FL system where the clients collaboratively train a shared Global Model without conceding their private data [5], [6].

B. DYNAMOS: Adaptive Microservice-based middleware

The *Dynamically Adaptive Microservice-based OS*, or **DYNAMOS**[1] for short, is a tool used for simulating data exchange scenarios in a distributed environment. DYNAMOS strives to establish a self-adaptive system capable of seamlessly incorporating policy, and archetypes, as well as functional and extra-functional adaptations, without manual intervention. Microservices [4] are composed of different configurations to serve different application goals in a distributed system.

C. FABRIC: International Testbed Infrastructure

The *Framework for Accelerated Built-In Resilience Infrastructure for Computing*, or **FABRIC** for short, is a cutting-edge research infrastructure created to support next-generation computing systems and networks. It offers a scalable, distributed environment equipped with high-performance computing resources, and integrates with containerization technologies such as Docker and Kubernetes. The testbed also features advanced networking capabilities [7]. Moreover, the *FABRIC Across Borders*, also known as **FAB**, extension connects the core North American network with global institutions, fostering international collaboration and speeding up scientific discovery.

III. Use Case Scenario

According to the EASA, the combination of AI-based predictive maintenance and the huge amount of fleet data available [8], could enable the early detection of potential engine failures, providing proactive maintenance [1] and potentially saving countless lives in the future of air travel. At a global level, the IDCA [2] recognizes the transformative potential of collaborative data sharing in the airline industry.

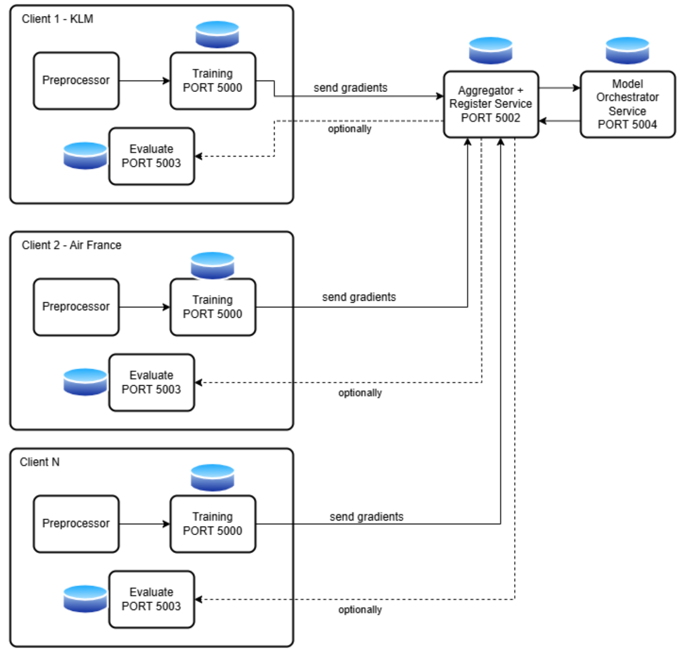

Fig. 1. Microservices-based implementation of collaborative airline data model training scenario with FL.

Motivated by this critical obstacle, we are using NASA's N-CMAPSS dataset [9], which was created to enhance predictive maintenance models for aviation engines [10], to demonstrate how airlines could leverage the power of FL and microservices, to collaboratively train a global ML-model that is able to accurately and automatically detect possible future engine failure [11]. We consider two major international airlines located at two different networking domains, using DYNAMOS to deploy a FL collaborative model training service based on the N-CMAPSS Dataset [9]. The architecture of the studied scenario can be seen in Fig. 3.

IV. System Architecture & Design Considerations

In this section, we provide a comprehensive overview of the architecture, highlighting the key design decisions that guided its development.

A. Microservices-based FL with DYNAMOS

In our proposed implementation of FL with microservices, there are two distinct types of agents:

First, the *Central Server* that is responsible for the management of the **Global Model (GM)**, that is collaboratively trained by the participating clients. The second type of agents are the *Clients*, they are parties that agree to participate in the collaborative FL and they are responsible for training and evaluating the shared global model locally with their own private datasets.

Figure 3 shows the microservice composition of the FL use case with two clients. In particular, Clients contain the **Training service** that takes as input the preexisting general model

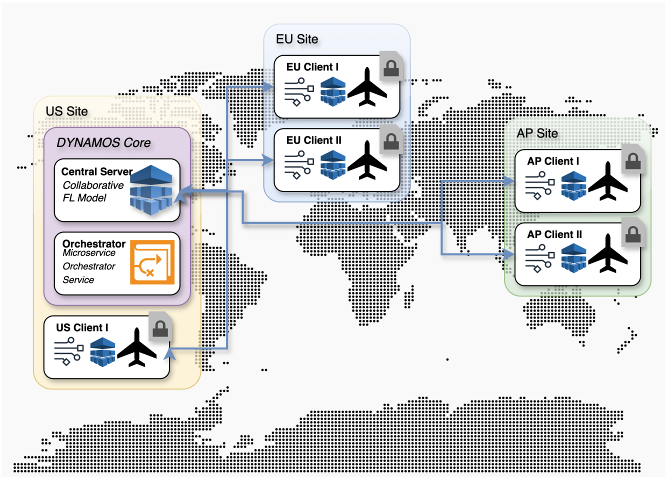

Fig. 2. Overview of a globally deployed microservices-based FL solution with DYNAMOS, tailored to the international aviation industry.

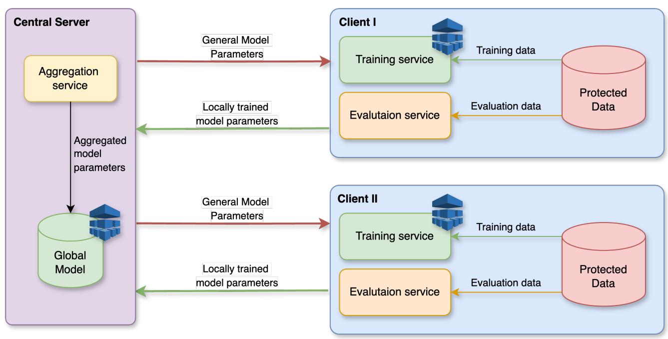

Fig. 3. Microservices implementation of the FL components and data exchange flows.

parameters and consecutively retrains it on the client's data. Moreover, the Client agent includes an **Evaluation service** that evaluates the GM after every cycle of FL to determine its local performance on the client's data The Central Server agent contains the **Aggregation service** that combines the locally trained models by the clients and updates the GM.

B. Containarization of FL components

The core of DYNAMOS is composed of the Orchestrator and the Central Server, as depicted in Figure 2. The Orchestrator is responsible for enforcing the policy agreement to certify compliance during the FL process and initializes the FL job with the appropriate parameters and participants. The Central Server component is a Trusted Third Party agent that is tasked with storing and distributing the GM to the clients and aggregating the locally trained models by the clients. Finally, the Clients are agents that are in agreement to collectively train the GM through FL.

Fig. 4 shows a sequence of calls during an iteration of collaborative training with FL with two clients. Initially, the data agreement agency (e.g. IDCA) makes the original request to DYNAMOS to begin the process. The request includes information about the clients to participate, training and aggregation configurations. Next, the Orchestrator processes and validates the request based on the contract and composes the corresponding microservice chain with an associated unique ID. Afterwards, the Central Server is triggered by the orchestrator's initialization message. The Central Server first initialises the Global Model according to the desired configuration and then sequentially instructs each client to train the GM with their data. When all clients have reported the updated parameters of the model back to the Central Server, the aggregation service is called to combine them into the updated version of the GM. Finally, the Central Server notifies all clients of the updated GM so that the clients can evaluate it on their own proprietary data and inform their decisions. The Aggregation Evaluation process can be repeated multiple times to improve the quality of the Global Model.

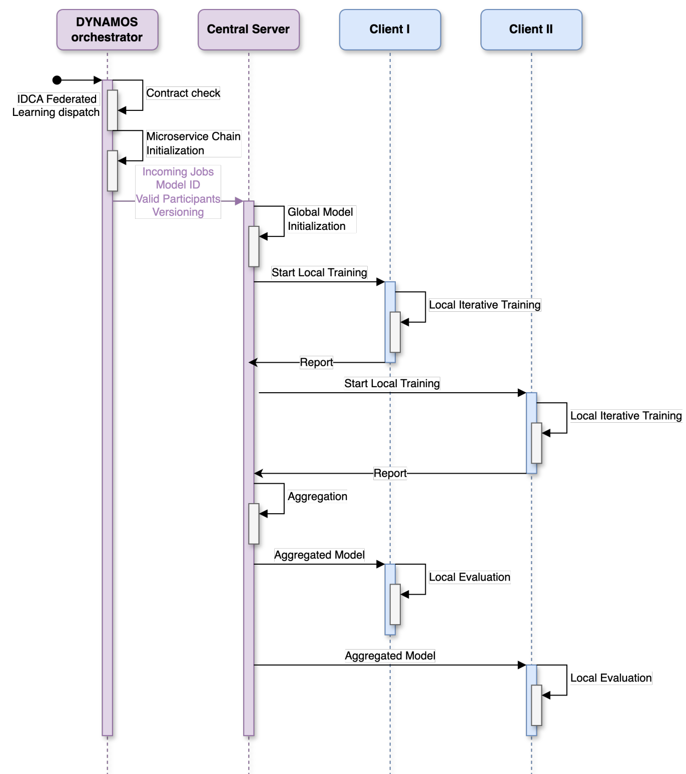

Fig. 4. Sequence diagram of one iteration of FL using DYNAMOS.

V. TECHNICAL EVALUATION

In this section we are presenting the evaluation scenario and the results extracted during the experimental evaluation.

A. Experimental Network Slice Overview

In this work, the experimental evaluation takes place in a custom testbed, designed and deployed in the FABRIC FAB network. We opted for a resource slice with 4 Virtual Machines (VMs) that spans between two sites, one located in the EU area and the other one at the USA, to simulate collaborative data sharing between international airlines.

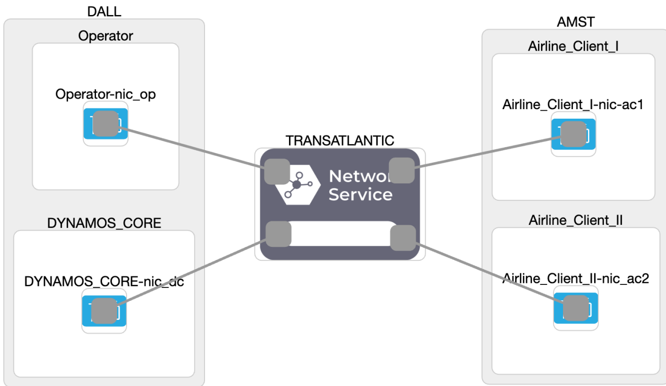

Fig. 5. FABRIC FAB slice configuration exported from the slice dashboard.

As depicted in figure 5, two VMs were dedicated to airline clients. One VM for the *DYNAMOS Core* Kubernetes Cluster and one for an *Operator* that was utilized to create performance testing scenarios. Two FABRIC Across Borders (FAB) international sites were chosen for this deployment, *DALL* the Equinix Data Center, located in Dallas, TX, US, and *AMST* the University of Amsterdam node, located at the Amsterdam Science Park, The Netherlands. The airline and DYNAMOS VMs provision 16GBs of RAM, 128GBs of storage and 4 Cores, while the operator machine occupies only 2 cores, 4GBs of RAM and 16GBs of disk storage. An IPv4 Layer-2 network was used to connect the two international sites. Docker and Kubernetes installations were performed to allow for containerized application life-cycle, orchestration and scaling by the DYNAMOS middleware.

B. Dataset

The dataset of choice used for this scenario is the NASA Turbofan Jet Engine Dataset (N-CMAPSS), as it reflects a critical collaborative training scenario among airlines. The N-CMAPSS dataset, an extension of NASA's earlier CMAPSS dataset [9], was created to enhance predictive maintenance models for aviation engines [10].

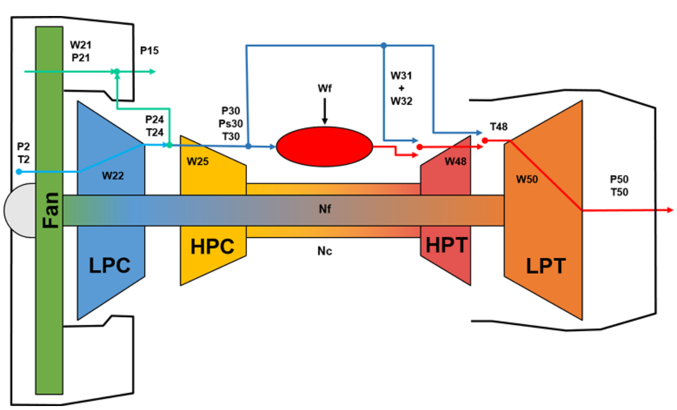

Fig. 6. Schematic representation of the CMAPSS model [10]

It provides high-fidelity simulations of turbofan engines, incorporating realistic flight conditions and component failures. The dataset contains multivariate time-series data from multiple flight cycles, covering climb, cruise, and descent phases. This comprehensive dataset is designed to reflect accurate engine degradation patterns, supporting research in predictive maintenance and prognostics [12]. Figure 6 provides a schematic representation of the simulated turbofan engine and its sensors as used in the N-CMAPSS dataset.

C. Scenario Results

The experimental topology of the deployed networking slice, exported from the slices tab in the FABRIC portal, can be seen in Fig. 5.

To demonstrate a realistic scenario of collaborative model training for the airline industry we are using a LinearRegression model from scikit-learn as the initial model parameters for aggregation. The original dataset, stored in HDF5 format, is partitioned into subsets. We assign ach subset to individual sovereign clients, that allows them to train local models independently on their specific data segments (dev_data and test_data). This partitioning approach facilitates parallel processing and scalable model training, leveraging the capabilities of each client without compromising data consistency. We conduct tests in two different scenarios:

- **Non-disagreement scenario:** Clients are willing to share data with one another and there are no conflicts.
- **Disagreement scenario:** Clients have conflicts and are not willing to share their data with one or more clients.

The performance metrics presented below in Table I were obtained using Apache JMeter [13].

Metric	Non-Disagreement	Disagreement
Average Response Time	1108.9 ms	1192.0 ms
Throughput	2.4 req/sec	0.8 req/sec
Max Latency	869.9 ms	601.5 ms

TABLE I
FL SCENARIO PERFORMANCE EVALUATION DEPLOYED WITH MICROSERVICES.

As we can see, the analysis shows significant variation between scenarios involving client disagreements and those without, which can be attributed to the inter-service communication and model aggregation in the context of FL between the microservice containers.

Finally, we provide a brief performance analysis between different types of collaborative training architectures, to justify the use of the DYNAMOS-based microservices approach. Each has strengths and weaknesses [14]–[17]. In Fig. 7, we compare key performance indicators such as the average response time, throughput, and maximum latency.

Although the performance of the microservices architecture shows notable differences between scenarios with and without client disagreements, the client-server architecture faces limitations in scalability because it requires the entire application to be inefficiently replicated. On the other hand, microservices excel in scalability by allowing individual services to scale independently based on demand. They optimize resource usage

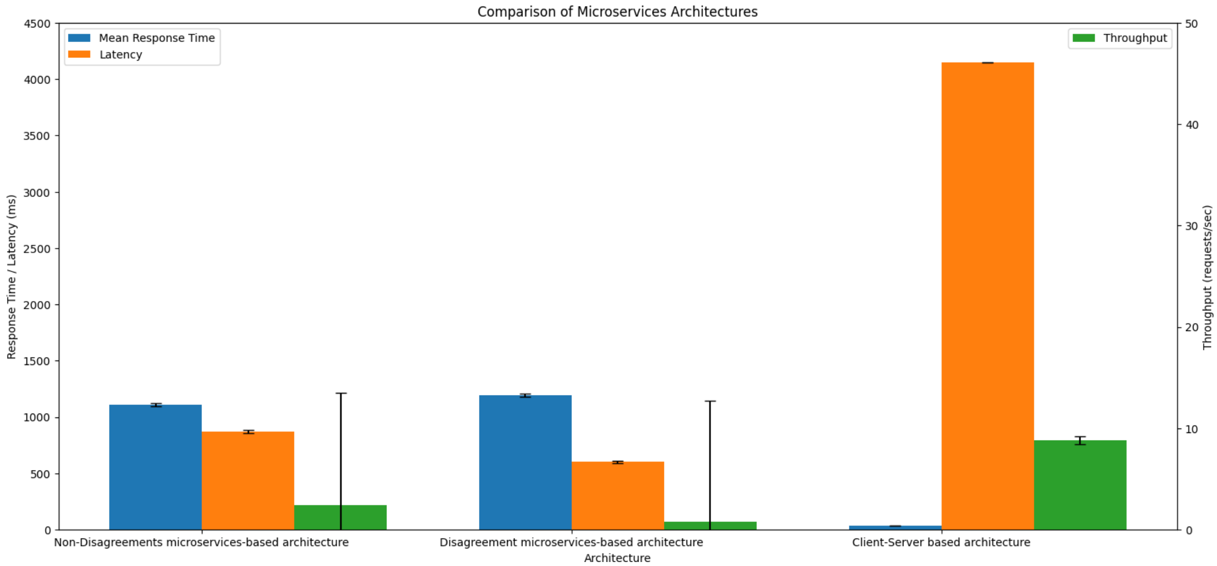

Fig. 7. Mean Response Time, Throughput, and Latency with Error Bars for Different Architectures.

and provide greater flexibility to adapt to changing requirements, despite the higher response times and lower throughput observed in the initial performance metrics.

VI. Discussion

The conjunction of this three-layer evaluation proves that international collaborative training is possible to foresee critical air-travel metrics is possible with current technology. This demo effectively addresses critical challenges, such as data partitioning, update management and adaptive learning, in a multi-domain scenario. The efficient use of microservices architecture thanks to DYNAMOS allows for scaling and automation between multiple parties. Finally, although the training is collaborative, via the advances in Software-Defined Networking (SDN), Network Function Virtualization (NFV) and FL, the data are always in secure, isolated and fully controlled environments, by their respective owners.

VII. Conclusion

In this paper, we leveraged NASA's Turbofan Jet Engine Dataset (N-CMAPSS) to demonstrate a viable solution to the obstacles international airlines face to collaboratively train models for predicting critical failures. By employing the FL technique and DYNAMOS middleware, we showcase how airlines can collaboratively train a global ML model to detect possible engine failures accurately, with current technology. Our deployment on the FABRIC FAB network validates the proposed approach, showcasing the ability to manage ad-hoc disagreements among FL participants dynamically in a real geographically distributed setting.

Acknowledgment

We would like to thank our KLM colleagues Jeroen Mulder and Asteris Apostolidis for their support with this research. This project was made possible with FABRIC and specifically FABRIC Across Borders, all funded by major NSF grants.

References

[1] EASA, "Easa artificial intelligence roadmap 2.0 - a human-centric approach to ai in aviation," May 2023.

[2] Data for aviation. [Online]. Available: https://dataforaviation.org

[3] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.

[4] J. Lewis and M. Fowler. (2014) Microservices: a definition of this new architectural term.

[5] M. Sabuhi, P. Musilek, and C.-P. Bezemer, "Micro-fl: A fault-tolerant scalable microservice-based platform for federated learning," *Future Internet*, vol. 16, no. 3, p. 70–70, Feb 2024.

[6] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, H. B. McMahan, T. Van Overveldt, D. Petrou, D. Ramage, and J. Roselander, "Towards federated learning at scale: System design," Mar 2019.

[7] [Online]. Available: https://portal.fabric-testbed.net/about/about-fabric

[8] J. B. Simon, D. Karkada, N. Ghosh, and M. Belkin, "More is better in modern machine learning: when infinite overparameterization is optimal and overfitting is obligatory," *arXiv (Cornell University)*, Jan 2023.

[9] S. Chatterjee and A. Keprate, "Exploratory data analysis of the n-cmapss dataset for prognostics," p. 1114–1121, Dec 2021.

[10] M. Arias Chao, C. Kulkarni, K. Goebel, and O. Fink, "Aircraft engine run-to-failure dataset under real flight conditions for prognostics and diagnostics," *Data*, vol. 6, no. 1, p. 5, Jan 2021.

[11] P. Korvesis, "Machine learning for predictive maintenance in aviation," Ph.D. dissertation, 11 2017.

[12] K. Goebel, J. Celaya, S. Sankararaman, I. Roychoudhury, M. Daigle, and A. Saxena, *Prognostics: The Science of Making Predictions*, 04 2017.

[13] A. S. Foundation, "Apache jmeter-apache jmetertm," 2019. [Online]. Available: https://jmeter.apache.org/

[14] N. Salaheddin and N. Ahmed, "Microservices vs. monolithic architectures [the differential structure between two architectures]," *MINAR International Journal of Applied Sciences and Technology*, vol. 4, pp. 484–490, 10 2022.

[15] K. Gos and W. Zabierowski, "The comparison of microservice and monolithic architecture," 04 2020, pp. 150–153.

[16] T. Salah, M. Jamal Zemerly, C. Y. Yeun, M. Al-Qutayri, and Y. Al-Hammadi, "The evolution of distributed systems towards microservices architecture," in *2016 11th International Conference for Internet Technology and Secured Transactions (ICITST)*, 2016, pp. 318–325.

[17] L. Rushani and F. Halili, "Differences between service-oriented architecture and microservices architecture," *International Journal of Natural Sciences: Current and Future Research Trends*, vol. 13, p. 30–48, Apr. 2022.