Aviation Data Analytics in the Cloud

Author: Esteban Ayala Arroyo
Advisor: PhD. Jeffrey Duffany
Computer Science Department, Polytechnic University of Puerto Rico

Abstract

Maintenance, Repair, and Overhaul (MRO) in commercial aviation refers to the practices and systems airlines use to ensure aircraft remain airworthy, efficient, and compliant with regulations over time. It's a broad term, covering everything from routine inspections to unscheduled repairs and full component overhauls. In daily operations, MRO shows up in ways that might seem invisible to passengers—like a component swap during a turnaround or the tracking of wear patterns through maintenance logs. Airlines rely heavily on reliability data, both historical and real-time, to anticipate failures before they happen and to schedule work without disrupting flight schedules. This data—collected from flight hours, shop findings, onboard sensors, even crew reports—feeds into reports that are often requested by customers or suppliers. Sometimes it's messy, incomplete, or needs stitching together from different sources. But when it works, it helps close feedback loops, improve parts sourcing, and meet contractual obligations.

Introduction

Maintenance, Repair, and Overhaul (MRO) operations are shifting with the times, blending traditional technical work with advanced digital tools [1]. These changes promise a lot, more reliability, faster turnaround, fewer unexpected failures. And yes, ideally, lower costs. But it's not just about installing a new platform and calling it progress. There's more to it. tools and platforms still depends on something deceptively simple: data. Reliable, complete, clean data. Without it, even the best algorithms produce noise. And let's be honest MRO records aren't always perfect. There are gaps, inconsistencies, even flat-out errors. Data mining can help fill in the blanks,

Problem

In the world of commercial aviation, maintaining aircraft safely and efficiently depends on far more than just skilled technicians and good equipment. Increasingly, it depends on data—specifically, the ability to gather, interpret, and act on it. That sounds straightforward, but in practice, it rarely is. MRO operations generate huge amounts of data: logs, sensor outputs, inspection notes, supplier documentation. Yet too often, this data arrives incomplete, inconsistent, or poorly formatted—especially from third-party sources.

Methodology

When working with aerospace data, especially for reporting purposes, there's often this gap—details like the tail number, engine model, or even simple things like flight cycles tend to go missing or just not provided but needed to make reliability calculations. It's not that they aren't out there. They are. But they're scattered, or perhaps not prioritized by the main data sources. That's where something like the Flight Radar API becomes really useful.

So here's what the project does, roughly speaking. It starts with ingestion. In this case, the source isn't a big internal database or a real-time sensor feed—it's an external API from *Rapid API*. We use that to pull enriched flight information: things like engine type, aircraft operator, ICAO airline code, flight duration, and the number of cycles. Especially important since they directly affect mechanical stress and cabin pressurization wear. A pressurization cycle is defined as the sequence in which the aircraft begins on the ground in an unpressurized state, becomes fully pressurized upon reaching cruise altitude, and then fully depressurizes upon descent and landing.

From there, we treat the data kind of like a pipeline. The first stage is just gathering and staging it—think of this as our version of Azure Data Factory. Simple, maybe a little messy. But it gets the data where it needs to be.

Then we process. This is where the logic comes in. Using Databricks, do some transformations. For instance, maybe we calculate the number of cycles based on timestamps and location patterns. Or we cross-reference tail numbers to enrich the record with engine specs that weren't part of the initial API call. This step is where the raw feed starts becoming something structured, something that actually means something.

After transformation, it needs a place to live—a layer where others (or systems) can query it. That's where Synapse Analytics would come in. In other cases, maybe it's a centralized database or warehouse optimized for flight reporting, audit logs, or performance summaries. It doesn't have to be perfect. Finally, visualization. Whether it's Power BI or Tableau, this step is all about surfacing that information. And not just for the sake of charts, it's about making the invisible visible

Results and Discussion

After integrating the Flight Radar API with a structured ingestion and transformation pipeline, the result is a solution that can improve how MRO data is processed—and, perhaps more importantly, how it is used. The most immediate result was speed. Maintenance records that previously took weeks to complete, due to fragmented or missing flight data, were now processed in near real time. Or close enough that it made a noticeable difference in scheduling and dispatch.

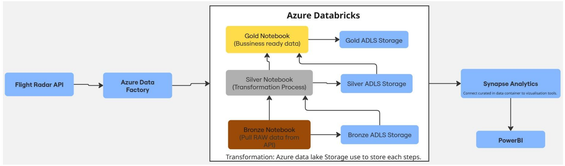

Figure 1. ETL Process

Results and Discussion

One of the more persistent pain points had been reconciling incomplete records. Tail numbers with no engine model. Flight durations without departure or arrival context. The automated solution, while not flawless, filled those blanks with a degree of consistency that had simply not been practical before.

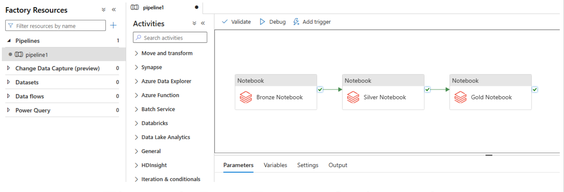

Figure 2. Data Factory Orchestration

Interestingly, the greatest improvement didn't show up in the dashboards at first. It showed up in the speed of deployment. Reliability calculations were cleared and returned faster, sometimes by hours, occasionally by a full day. That might not sound dramatic, but over a month, across a fleet, that's a tangible operational gain. One that mattered more than we initially expected.

Conclusions

This project wasn't about building the most advanced model or deploying cutting-edge AI. It was about solving a problem that shows up in the day-to-day, missing data, fractured records, and the ripple effects that follow when you can't fully trust your inputs. By building a pipeline that ingests, enriches, and surfaces MRO-relevant data in a way that's structured and timely, we weren't chasing perfection—we were chasing usability. And that made all the difference.

Future Work

There's still a lot more that can be done. This project focused mainly on cleaning and structuring the data—getting it into a place where it's actually usable. That alone made a difference. But looking ahead, the next logical step is to move from *descriptive* insights to *predictive* ones. Not just seeing what happened, but anticipating what's likely to happen next. But to get there, the models need more than just clean inputs. They need volume, context, and ideally, feedback loops. That means integrating additional data sources—shop findings, maintenance deferrals, even technician notes—alongside flight and sensor data.

Acknowledgements

I would like to acknowledge Dr. Jeffrey Duffany, professor at the Computer Science & Engineering department for guiding me and providing feedback throughout this work and Joann Casillas for the editing.

References

[1.] Anant Sahay (2012). Leveraging Information Technology for Optimal Aircraft Maintenance, Repair and Overhaul (MRO) retrieved March 15. From Book ISBN: 9781845699826.

[2.] Asteris Apostolidis (2020) Aviation Data Analytics in MRO Operations: Prospects and pitfalls. Retrieved April 5, 2025. From: IEEE Xplore: https://ieeexplore.ieee.org/document/9153694

[3.] Maurice Pelt et al. (2019) Data analytics case studies in the maintenance repair and overhaul (MRO) industry. Retrieved March 26,2025. From Matec Conference: https://www.matec-conferences.org/articles/matecconf/abs/2019/53/matecconf_easn2019_04005/matecconf_easn2019_04005.html