Things to Consider When Building an Airflow Service
About Us

Pete Dejoy
SVP Product, CoFounder

Viraj Parekh
Field CTO, CoFounder
About Us
About Us
A data engineering team uses Airflow...
But then more users come along...
...some of whom are different personas

**Analytics Engineer**
- SQL queries, dbt jobs
- “I need this report on monthly active users to be refreshed daily!”

**Data Engineers**
- Data ingestion, ETL, Schema Maintenance
- “I need to enrich our MAU tables with data from our legacy system!”

**Data Scientists**
- Notebooks, models, R, Python
- “I need my model to be trained on the latest data!”
You want to avoid this...

Can you make these reports run on time every day?

Can you productionize my model for me?

Ingestion, ETL, schema maintenance

“Can you all help me out?”

Data Analysts

SQL Queries, dbt jobs

Data Engineers

Python, Notebooks, ML Models

Data Scientists
…and build towards this.

Data Analysts: SQL Queries, dbt jobs

Data Engineers: Data ingestion, legacy workloads

Data Scientists: Python, R, Notebooks, Models
To build a robust Airflow service...

Three biggest challenges to enabling Airflow for all.

**Infra Mgmt**
How do you manage the infrastructure abstractions to deliver a reliable service?

**Workload Compute**
How do you give end users access to compute specialized for their use case?

**DevEx**
How do you deliver a world-class developer experience for different kinds of end users?
Infrastructure Management
There are a lot of Airflow services to build your service around...
...but they all differ in:

- **Tenancy Abstractions** - how do you scale across teams?
- **Feature set** - what Airflow features are exposed? What can you modify?
- **Cost** - What’s the pricing model? How do you justify ROI?
Many Approaches to Multitenancy

- Leverages Astronomer to run multiple Airflows
- Teams leverage their own workload patterns
- Cost savings resulted from optimizing workloads

- Monolithic architecture
- More opinionated data processing patterns
- Tracks cost with GKE native features
Workload Compute Needs
Practitioners and Infra Needs

<table>
<thead>
<tr>
<th>Practitioner</th>
<th>Skillset</th>
<th>Use Case</th>
<th>Infra Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analytics Engineers</td>
<td>SQL, dbt, BI Tools</td>
<td>“I need this <strong>report</strong> on MAU to be refreshed daily.”</td>
<td>- Transformation-oriented workflows</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Compute is generally offloaded to a DWH</td>
</tr>
<tr>
<td>Data Engineers</td>
<td>Python, SQL, Schema design, git</td>
<td>“I need to <strong>ingest some data</strong> from my app db to my data warehouse.”</td>
<td>- Need for ephemeral storage and disk space</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Need to manage python dependencies</td>
</tr>
<tr>
<td>Data Scientists</td>
<td>Pandas, Notebooks, R</td>
<td>“I need to write &amp; deploy a <strong>predictive model</strong> on MAUs.”</td>
<td>- Every compute need imaginable</td>
</tr>
</tbody>
</table>
Airflow has rich abstractions for customizing workload compute...

**Containers**
Operators for Kubernetes, Docker, ECS, ACI, and more for running pre-existing images

**Executors**
Celery, Kubernetes, and hybrid style executors to configure interface between user code and underlying infrastructure
Airflow has Abstractions for workload compute:

**Containers**

- GitLab
  - Run all their tasks in KubernetesPodOperators to handle dependencies + infra specifics
  - Links (1) (2)

**Executors**

- shopify
  - Executor optionality allows teams to choose the right tool for their workload profile.
  - Links (1) (2)

https://shopify.engineering/lessons-learned-apache-airflow-scale
ML Launcher – to containerize task execution. ML Launcher integrates compute backends like Sagemaker, Databricks, and Snowflake to perform container runs and meet the unique hardware requirements for ML such as GPUs, instances with large memory, and disks with high IO throughput. This design choice enables MLEs to develop and deploy pipelines without worrying about Airflow runtime and allows us to scale easily to hundreds of DAGs (Directed Acyclic Graphs) with thousands of tasks in a short period.

Airflow @ Instacart
Compute Abstractions → Interfaces + DevEx
Build your devex based on the needs of your end user!
<table>
<thead>
<tr>
<th>Practitioner</th>
<th>Skillset</th>
<th>Use Case</th>
<th>Interface of choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analytics Engineers</td>
<td>SQL, dbt, BI Tools</td>
<td>“I need this report on MAU to be refreshed daily.”</td>
<td>BI Tools, SQL tools, dbt</td>
</tr>
<tr>
<td>Data Engineers</td>
<td>Python, SQL, Schema design, git</td>
<td>“I need to ingest some data from my app db to my data warehouse.”</td>
<td>IDEs, Object oriented Python, SQL tools, some Terraform</td>
</tr>
<tr>
<td>Data Scientists</td>
<td>Pandas, Notebooks, R</td>
<td>“I need to write &amp; deploy a predictive model on MAUs.”</td>
<td>Notebooks, Python, Pandas, scripting</td>
</tr>
</tbody>
</table>
There are a ton in the community that exist:

https://airflow.apache.org/ecosystem/
TL;DR:

- Use a managed service, but know they’re not all built the same
- Different personas are going to have different compute & interface requirements
- Focus on developer experience, but if you’re building your own Airflow DSL, make sure you know what you’re getting into.
(After) Party Under the Stars

Wednesday, September 20th
6:30pm-10:00pm

The Sheraton Centre
123 Queen St W
(7 min walk)

RSVP Now
Let’s flow together

Workshop
Get Airflow Certified

Thursday, September 21st
12:00 pm in Trinity 4

Marc Lamberti
Head of Customer Education
at Astronomer