



Do you trust
Airflow with your
money? We do*!

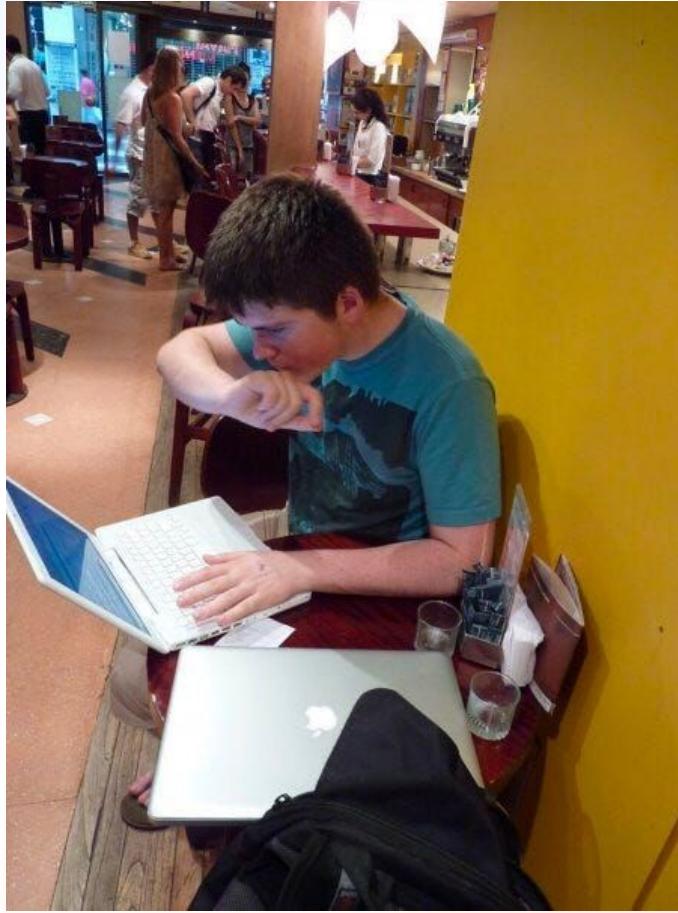
3.0

Sabrina Liu

Senior Software Engineer, Stripe

**In 2010, Stripe was a
hole in the wall**

stripe



John Collison, cofounder of Stripe, in 2010

The backbone for global commerce

Stripe makes moving money as easy and programmable as moving data. Our teams are based in offices around the world and we process hundreds of billions of dollars each year for ambitious businesses of all sizes.

500M+

API requests per day, peaking at 13,000 requests a second.

99.999%

historical uptime for [Stripe services](#).

90%

of U.S. adults have bought from businesses using Stripe.

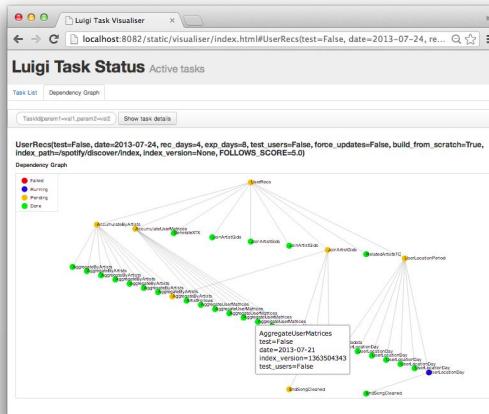
135+

currencies and payment methods supported.

Mid-2010s

2017 - present

What are data pipelines??



A screenshot of the Apache Airflow interface showing the DAGs page. The top navigation bar includes links for Airflow, DAGs, Data Profiling, Browse, Admin, Docs, and About. Below the navigation is a section titled "DAGs" with a table listing five DAGs: "example_bash_operator", "example_branch_dop_operator_v3", "example_branch_operator", "example_xcom", and "latest_only". The table columns are "DAG", "Schedule", and "Owner". The "example_bash_operator" DAG has a schedule of "0 0 * * *" and is owned by "airflow". The "example_branch_dop_operator_v3" DAG has a schedule of "/1 * * * *" and is owned by "airflow". The "example_branch_operator" DAG has a schedule of "@daily" and is owned by "airflow". The "example_xcom" DAG has a schedule of "@once" and is owned by "airflow". The "latest_only" DAG has a schedule of "4:00:00" and is owned by "Airflow". At the bottom of the page, there is a pagination control with buttons for '<', '< 1', '>', and '>>'.

Airflow at Stripe in 2025

15,000

unique Task classes

180,000

unique Tasks instantiated

10 million

daily task instance executions

2 Airflow clusters

Why Airflow 1, 2, or 3 out of the box don't solve our problems

We have an **UberDAG** (not you, Dara)

I want to reuse code across the **monorepo** like a
good engineer

Just run my workload

Let my task live for **3 days**

What we've built

Airflow but easier

Low-code + no code
orchestration on top of
Airflow

Local testing

User Scoped Mode (USM)

Multitenancy

Project-level isolation and
reliability on a single Airflow
cluster

Airflow but easier

The screenshot shows a user interface for managing pipelines. At the top, there are two sections: "Starred Projects" (Pipeline projects you've starred) and "Team Projects" (Projects owned by data-transformation). Below these are search and status filters. A main table lists pipelines with columns for Name, Status, and Description. The pipelines listed are:

NAME	STATUS	DESCRIPTION
demo-tutorial-project	active	This is a demo
demo-pipeline-studio	active	A demo
data-transformation-metrics	active	Operational and team metric
my-new-pipeline-dmelchor-v2	active	A new pipeline
recent-enrichment-count-test	draft	Test pipeline draft
ncp-data-transformation	active	Automatically migrated NCP
data-transformation-airflow-v2	active	All tasks that will only run in
data-transformation-canaries	active	Data Transformation pipeline
recent-enrichment-count-test	active	Test pipeline

At the bottom, there is a "View: 10" dropdown.

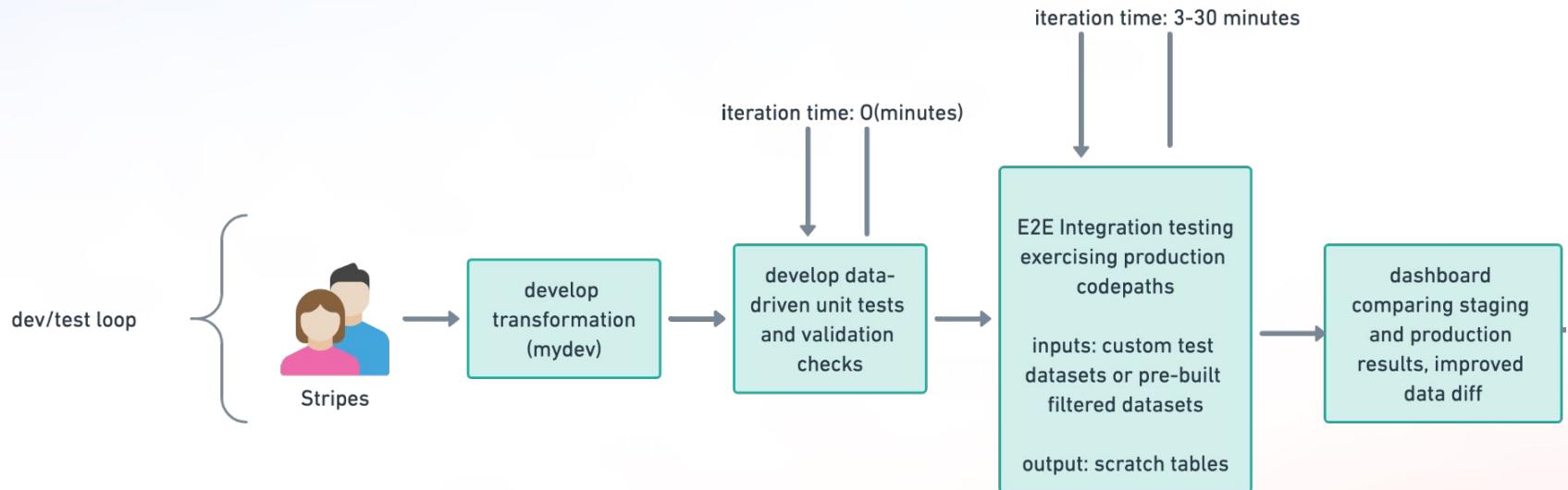
Simple Pipelines:

yaml-based task
authoring using
SparkSQL

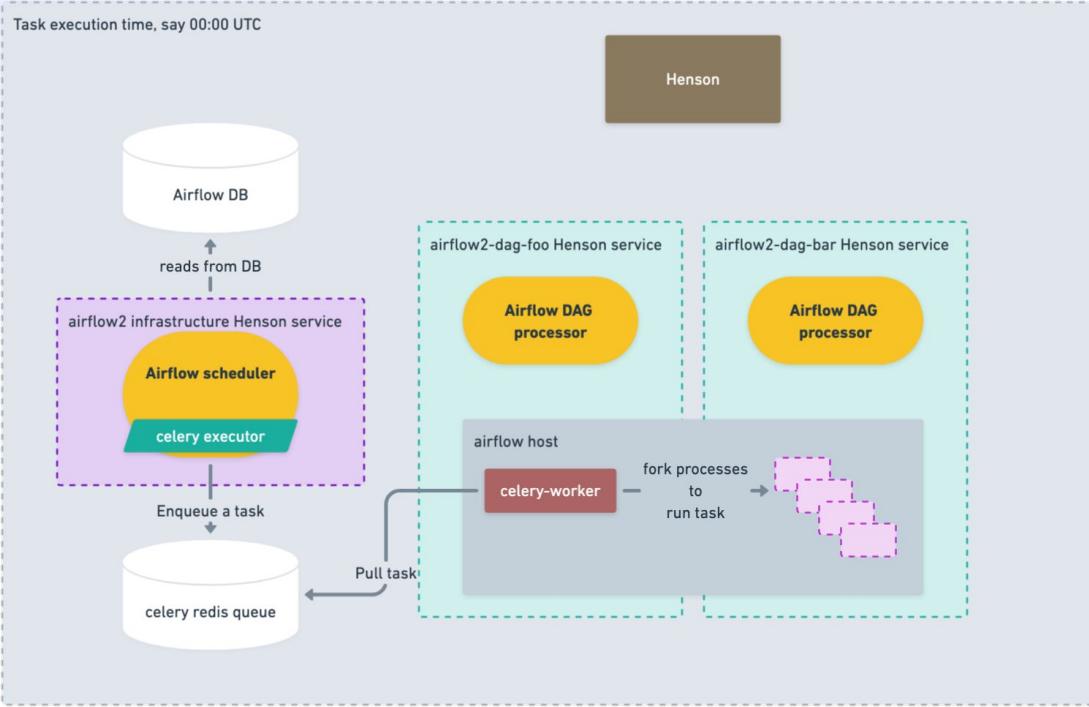
Pipeline Studio:

no code task
authoring using SQL

User Scope Mode



Key: production inputs, non-production outputs



Improvements

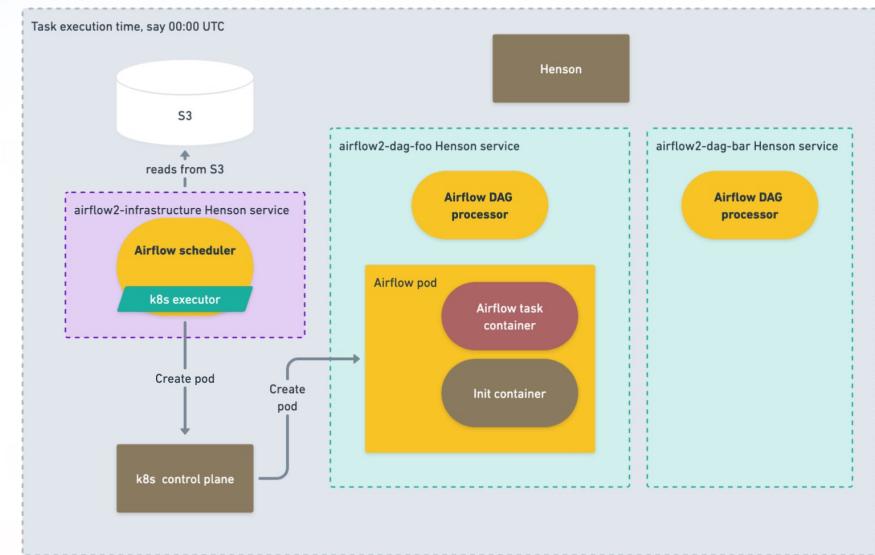
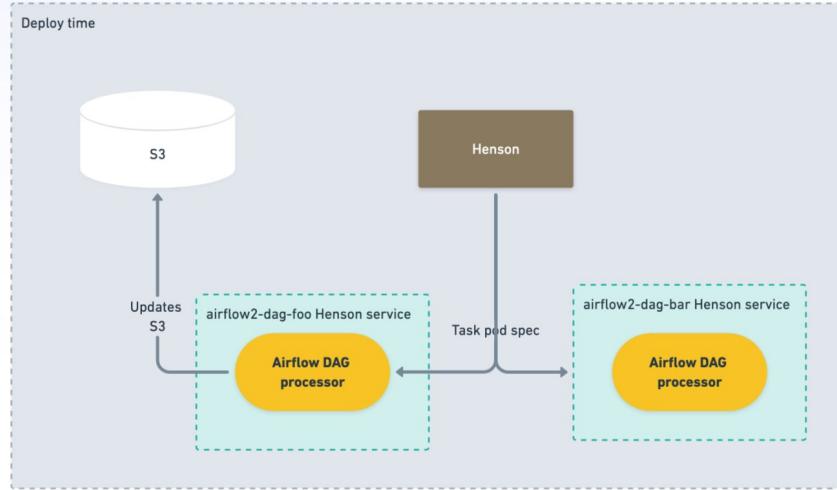
- 1 project : 1 service : 1 dag processor : 1 owning team

Weaknesses

- Hard to scale deployments
- Manual Celery shard management
- Shared workload identity

Step 1 to multitenancy with EC2 and Celery

Multitenancy with KubernetesExecutor



Independent execution and configuration

Customizations with the KubernetesExecutor

- **[scalability]** Multi-shard support
 - One executor per Kubernetes shard
 - We add a `cluster_context` attribute to the `kube_client`
 - Dedicated Kubernetes shard for Airflow workloads
- **[compliance]** SOX controls for workloads that touch financial data
 - Separate Kubernetes namespaces
 - Lifecycle management for long-running, stateful workloads
- **[efficiency]** Semi-managed compute
 - API for requesting CPU and memory
 - Automatic bin-packing for short-lived tasks

... and much more!

**... but we haven't
won yet!**



Sabrina Liu

Senior Software Engineer

LinkedIn:



Sharadh
Krishnamurthy

Engineering Manager

LinkedIn:

