Lyft Talks #4

Orchestrating big data and ML pipelines at Lyft

MODERATOR

Olexii Verkunich

Technical Recruiter

overkunich@lyft.com

Join the Ride!

AND GET $5K SIGN-ON BONUS

(Applicable for ENG openings in Minsk only. Must apply and/or get a job offer between Nov 18 and Jan 30 to qualify)

Constantine Slisenka
SWE (Software Engineer)

cslisenka@lyft.com

Agenda

1
Use-cases
big data, ML

2
Big data at Lyft
scale, ecosystem

3
Orchestration
Airflow, Flyte

4
Flyte live demo

?
Q&A

Use-cases

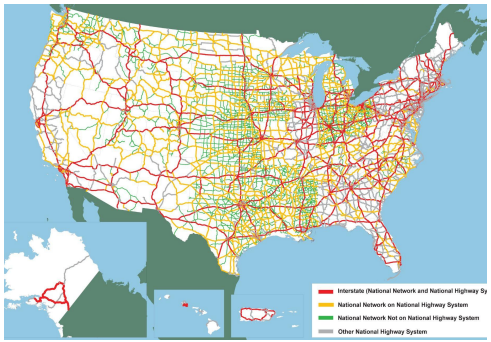
#big data
#ml

Keeping map data accurate and fresh

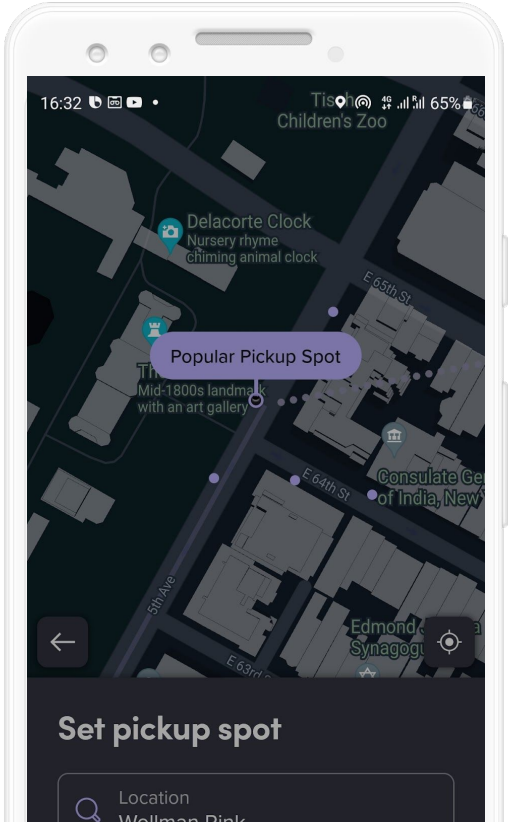
We have high quality curated map data from various sources including open data like OpenStreetMap

We improve the map to make it more accurate by processing imagery and gps telemetry for recognizing objects on the road like road signs, closures, traffic cones

The impact is a more optimal routes calculation, better ETA, and trip cost estimation



Calculating and suggesting pick up spots



We have previous pick up history

We recommend users best nearby options for pickups

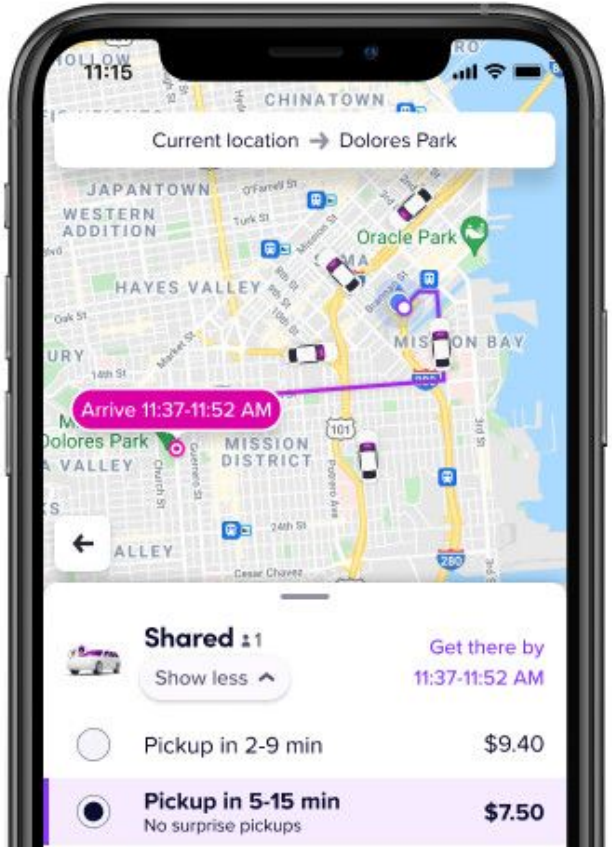
The impact is a better user experience with less driver friction: pick up in the optimal locations, more rides due to fewer cancellations

Detecting missing and inaccurate destinations

We have anomalies in ride

We detect when our data shows patterns suggesting inaccurate destinations

The impact is a better user experience, users are able to effectively find their destinations, more rides



Route calculation, ETA/price estimation

We have rich map data and telemetry from driver mobile devices

We generate real-time speed profiles and build a probabilistic model of routes

The impact is better routes, ETA and price estimates

Forecasting of traffic, demand, and supply

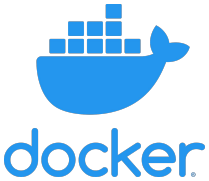
We have driver location data, information about events, rides history

We forecast demand and supply, get understanding of market balance

The impact is efficient pricing to serve more rides, more informed decisions around which incentives to give drivers (i.e. bonus zones)

Big data at Lyft

#ecosystem
#scale



kubernetes

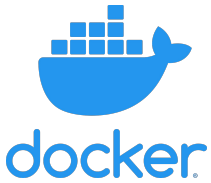
infrastructure

compute engines

Flink

beam

stream processing



infrastructure

compute engines

stream processing

development, reporting

storage and metadata

orchestration, ETL

analytical events

?

LAST MONTH

?

DATA IN S3

?

LAST MONTH

?

JOBS LAST MONTH

?

CONTAINERS LAST MONTH

ETL

?

PIPELINE RUNS

?

TASK EXECUTIONS

analytical events

8 773 789 938 145

LAST MONTH

amazon S3

?

DATA IN S3

?

LAST MONTH

?

JOBS LAST MONTH

?

CONTAINERS LAST MONTH

ETL

?

PIPELINE RUNS

?

TASK EXECUTIONS

analytical events

8 773 789 938 145

LAST MONTH

50PB

DATA IN S3

+400GB

LAST MONTH

?

JOBS LAST MONTH

?

CONTAINERS LAST MONTH

ETL

?

PIPELINE RUNS

?

TASK EXECUTIONS

analytical events

8 773 789 938 145

LAST MONTH

50PB

DATA IN S3

+400GB

LAST MONTH

376K

JOBS LAST MONTH

5M

CONTAINERS LAST MONTH

ETL

?

PIPELINE RUNS

?

TASK EXECUTIONS

analytical events

8 773 789 938 145

LAST MONTH

50PB

DATA IN S3

+400GB

LAST MONTH

376K

JOBS LAST MONTH

5M

CONTAINERS LAST MONTH

ETL

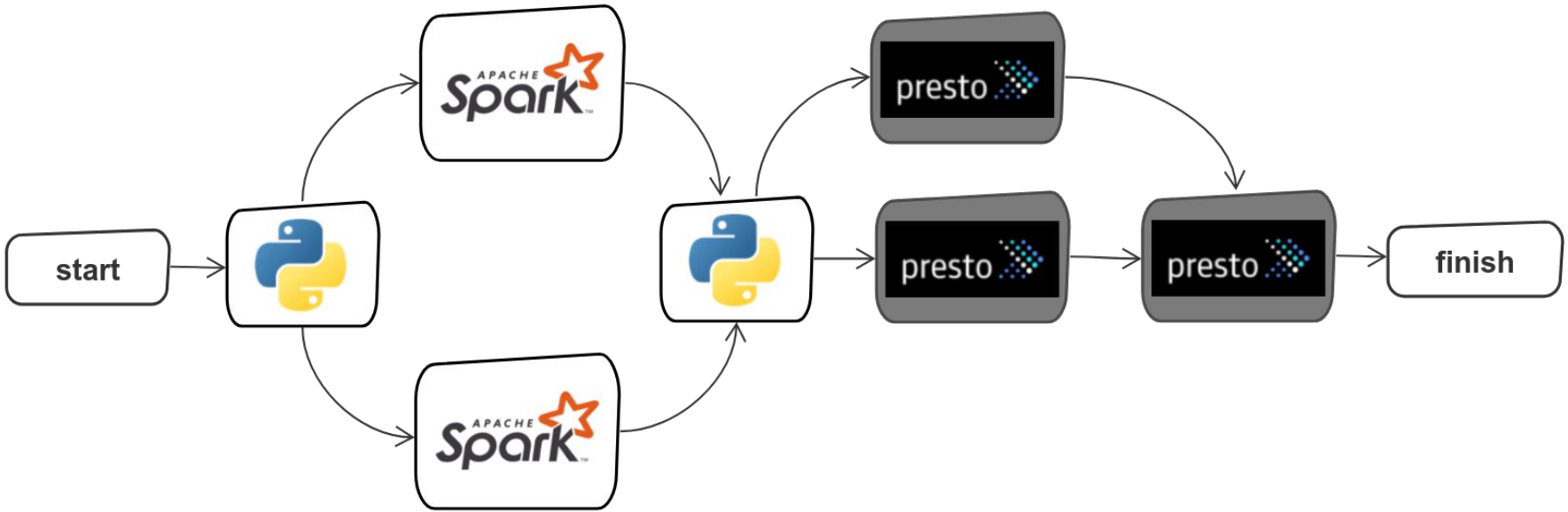
650K

PIPELINE RUNS

24M

TASK EXECUTIONS

Orchestration


start
APACHE Spark
APACHE Spark
presto
presto
presto
finish

Orchestration engines at Lyft

- Run pipelines (scheduled and ad-hoc)
- Provide python DSL
- Provide integrations with third party systems (hive, presto, spark, ...)
- Not compute engines
- Good for batch execution
- Not for data streaming

Orchestration engines at Lyft

Apache Airflow

- Run pipelines
- Provide integrations with third party systems (hive, presto, ...)
- Not for data streaming

Flyte

- Good for batch execution
- Not for data streaming

The devil is in the details

What is the difference between Flyte and Airflow?

Why  created Flyte?

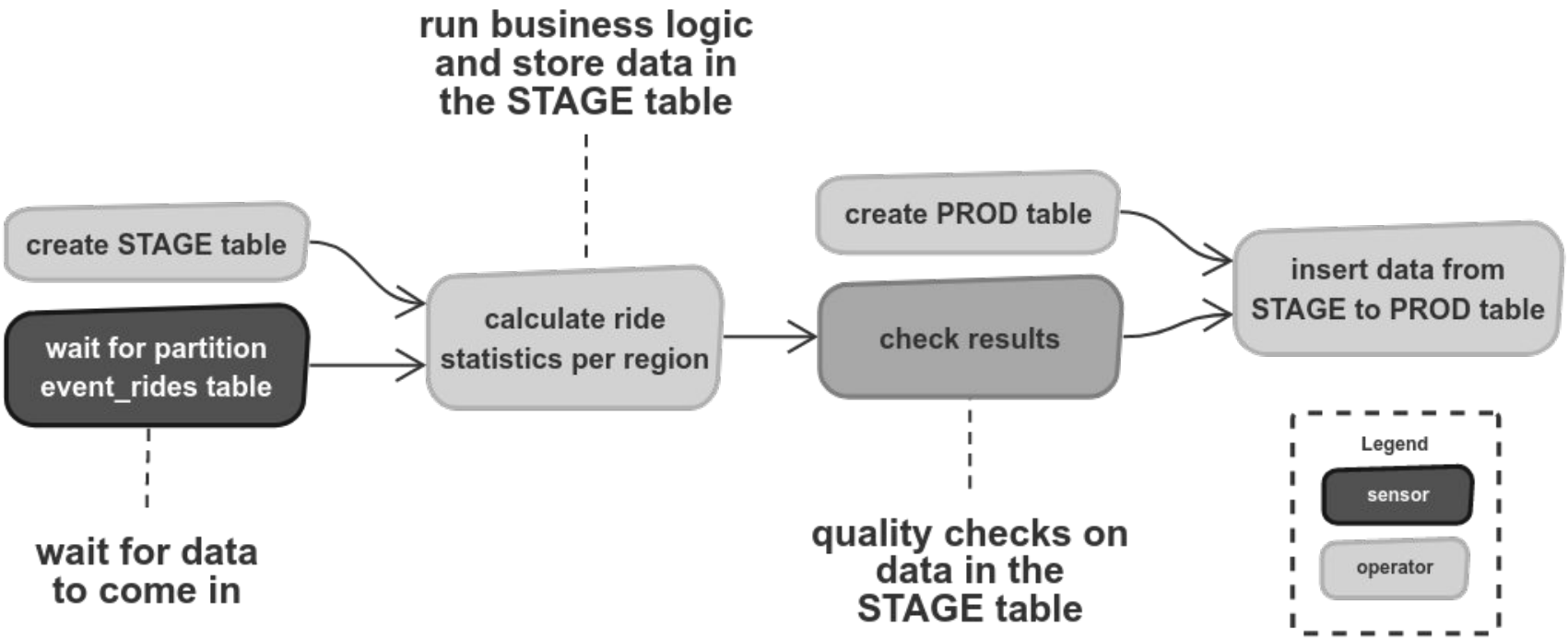
Why do we use both?

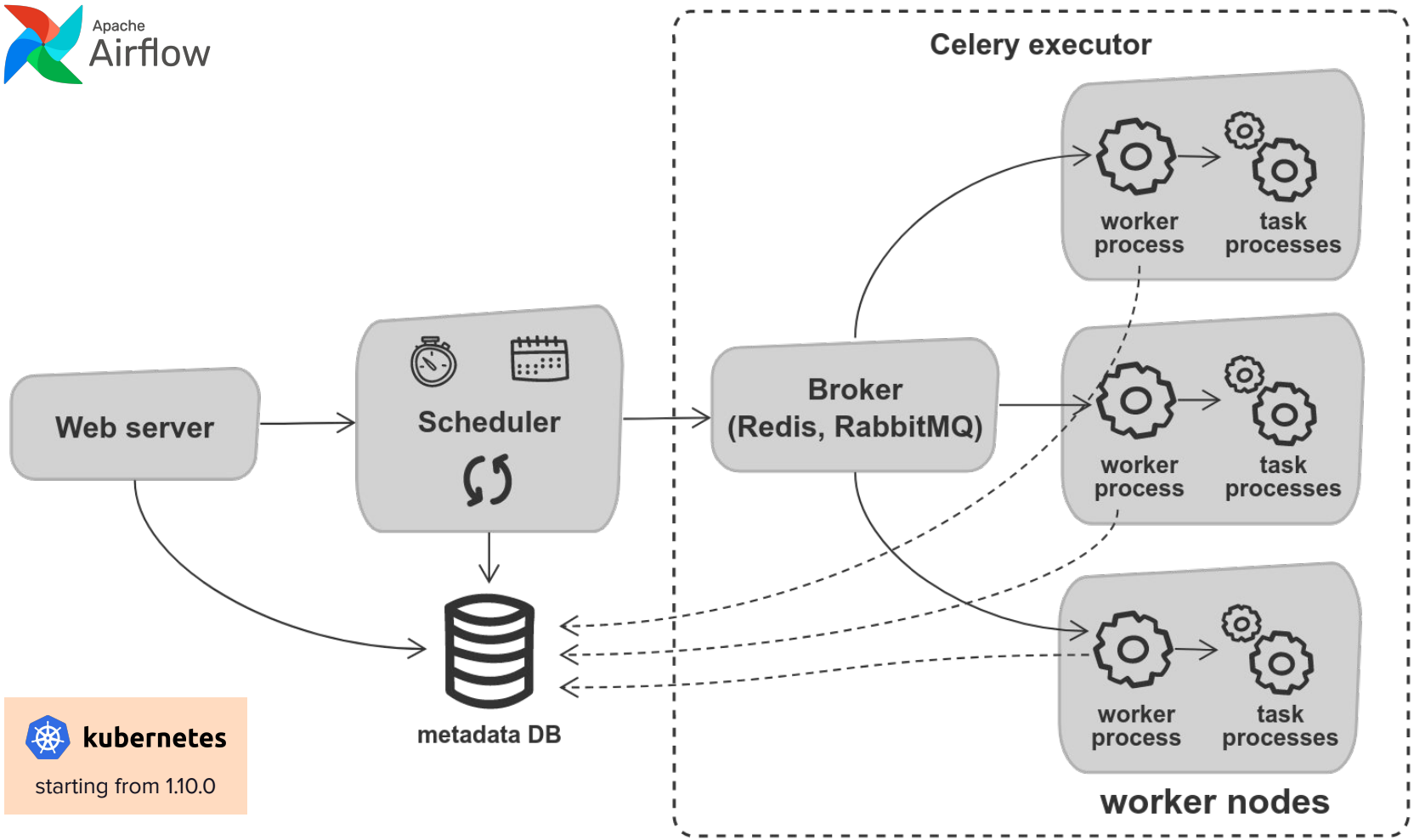
Should I use Flyte or Airflow for my project?

Apache

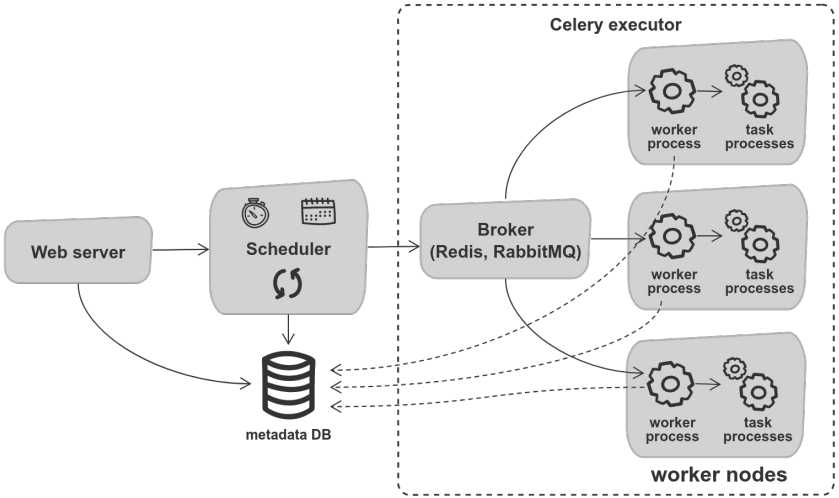
Airflow

Airflow DAGs




Apache
Airflow
Celery executor
worker
process
task
processes
Web server
Scheduler
Broker
(Redis, RabbitMQ)
worker
process
task
processes
worker
process
task
processes
metadata DB
kubernetes
starting from 1.10.0
worker nodes

Apache Airflow



- Quick and simple to start
- Many integrations (operators)
- Good support for sensor tasks
- Monolithic
- Fixed set of workers
- Does not manage infrastructure

No multi tenancy

- No environment separation: DEV, STAGE, PROD
- Impossible to set up a custom libraries and dependencies per DAG

Limited functionality

- No versioning of DAGs (no way to compare outputs of version A vs B)
- No caching of task results (Airflow is not data aware)

Monolithic scheduler

- Centralized scheduler becomes a bottleneck

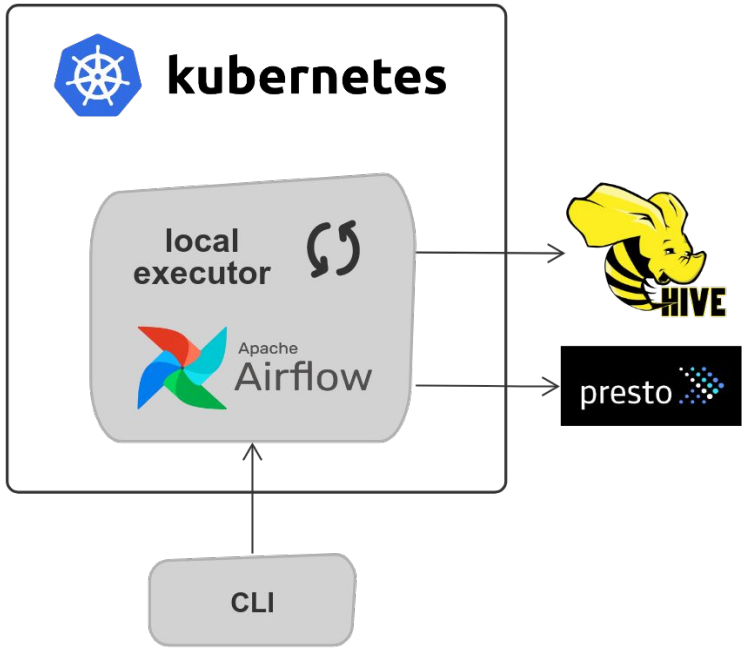
No resource management

- Heavy tasks may overwhelm worker
- Impossible to set resource quotas per task like max memory/CPU

TARS

- Airflow development environment for testing and backfilling
- Kubernetes pod with ETL software and libraries and CLI tools

(TARS is also Interstellar movie robot)



Apache Airflow

Good tool for classic ETLs

using a **standard set of operators** and

orchestrating **third-party systems** when

custom environment and **multi tenancy**

are not required


Flyte

Flyte

Nov, 2016
Flyte V0 built for
ETA team at Lyft

Nov, 2019
Flyte was open sourced
at Kubecon!
flyte.org

Q2 2020
Spotify and Freenome join
Flyte as collaborators

Q1 2021
Flyte was donated to LF AI &
Data Foundation
Union.ai started

LF AI & DATA
INCUBATION PROJECT

Union.ai

Q3 2021
15 collaborator
organizations
100+ contributors
Spotify contributes
flytekit-java

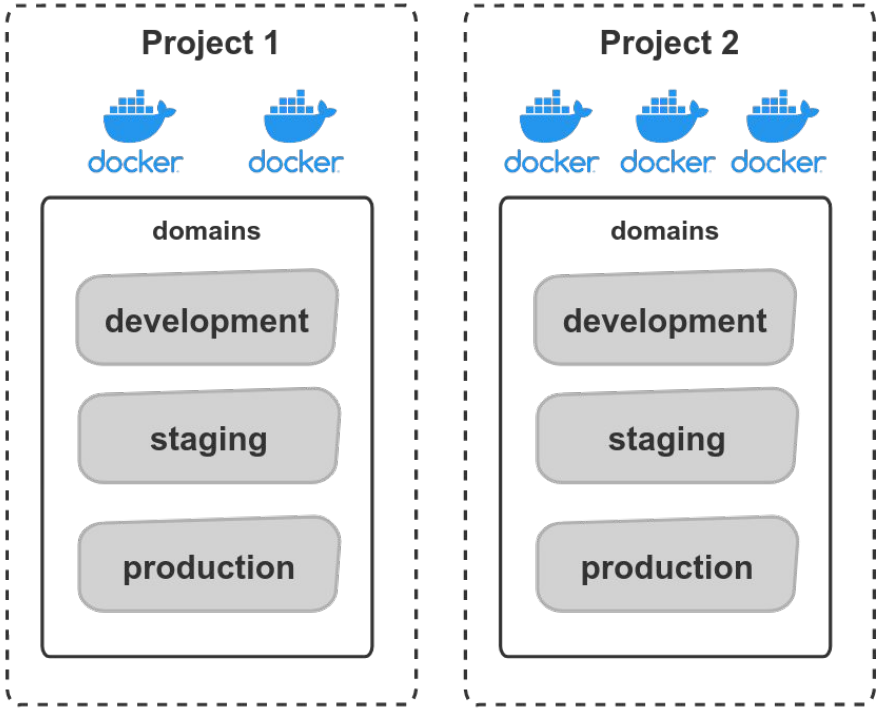
Multi tenancy

Flyte

Workspace is organized into **projects**

Projects or individual tasks have different **environment**

Projects are organized into **domains**: development, staging, production

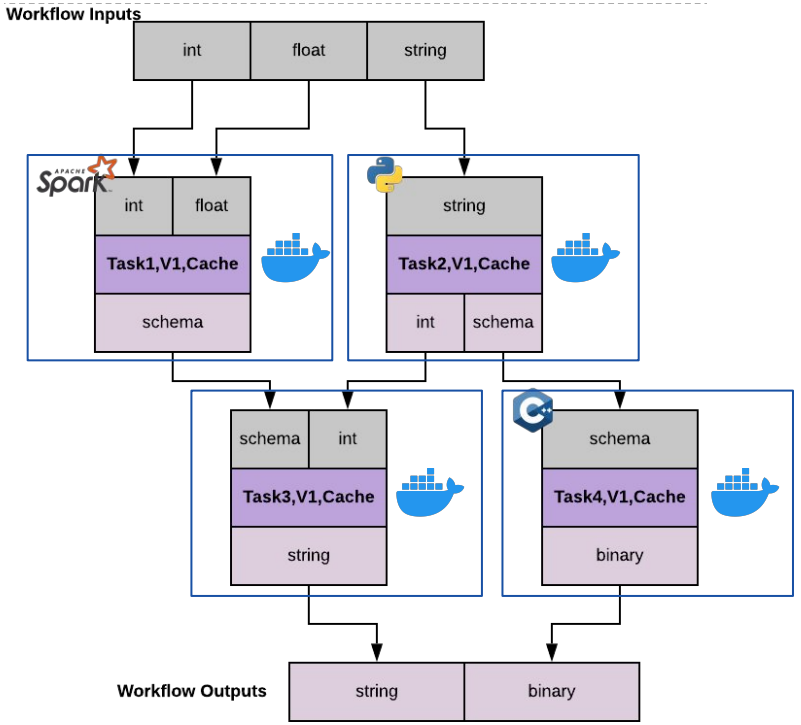


Flyte workflows

Language agnostic: can be written in python and java, any docker image can be a task

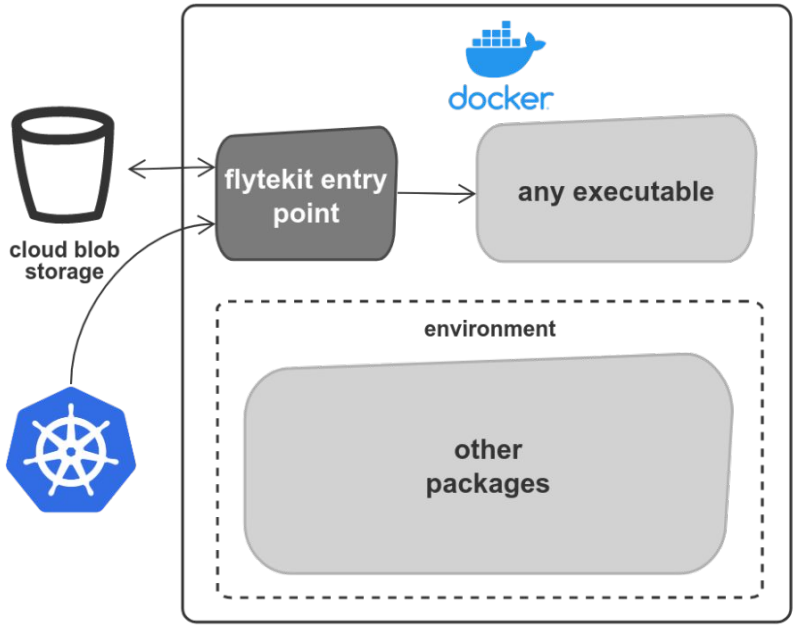
Versioned: each version is a separate docker image

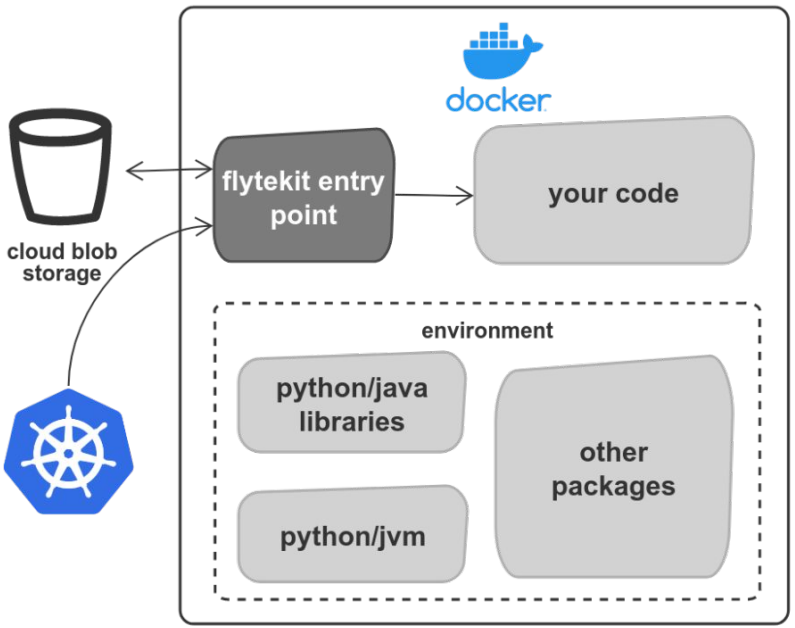
Data aware: strong typing for inputs and outputs, Flyte executes tasks based on data dependencies, results can be cached

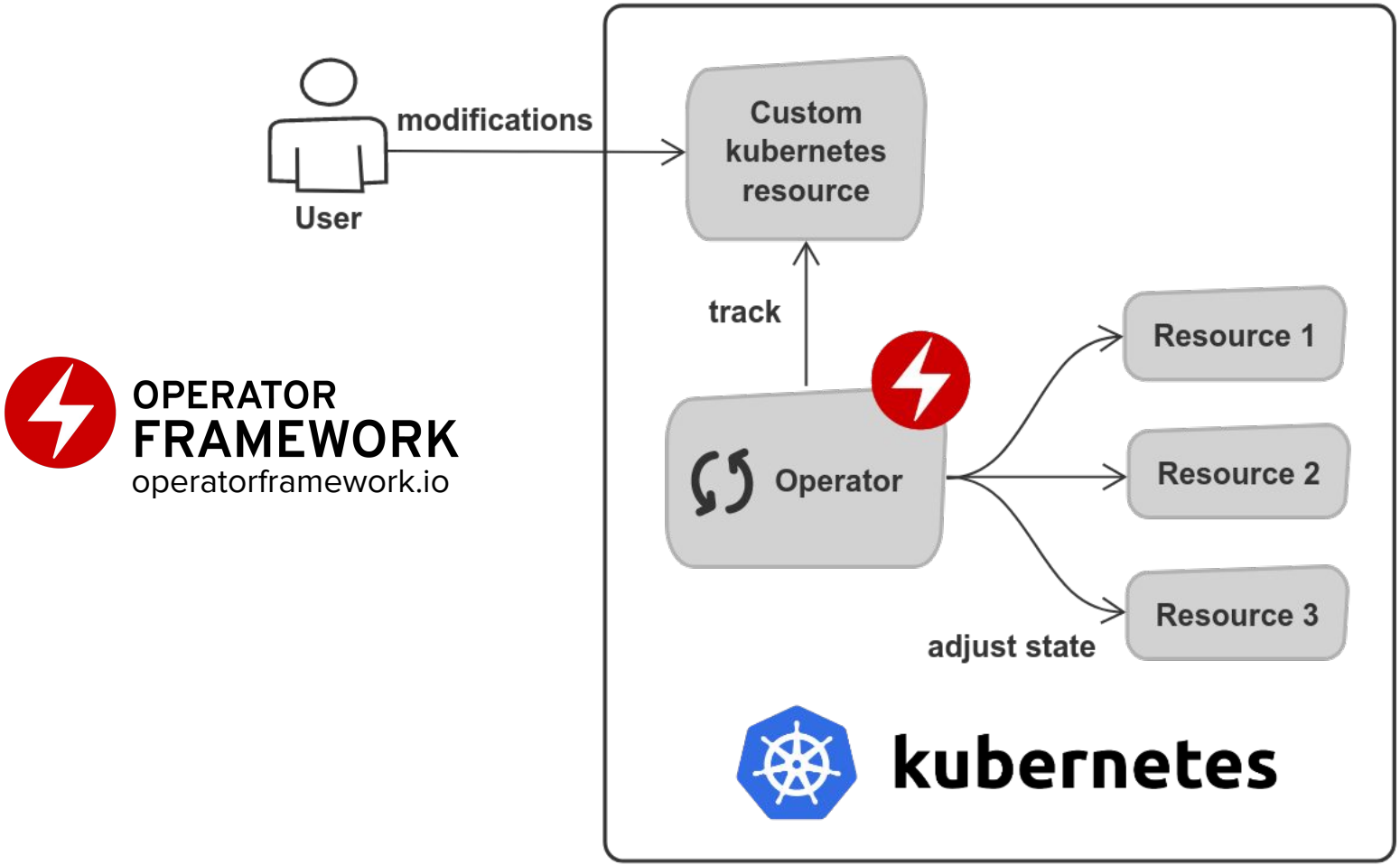


Flytekit (Flyte SDK)

Flyte

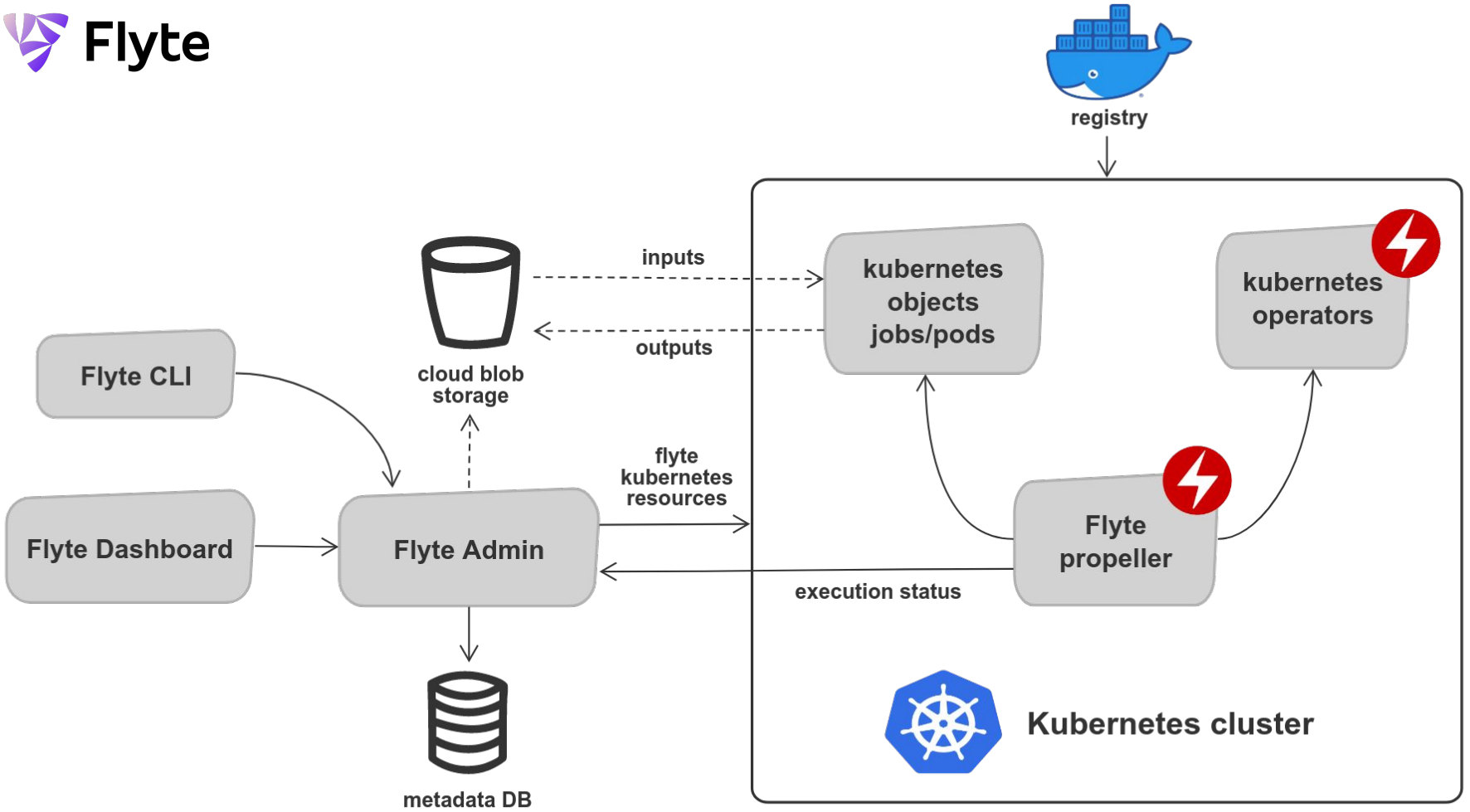




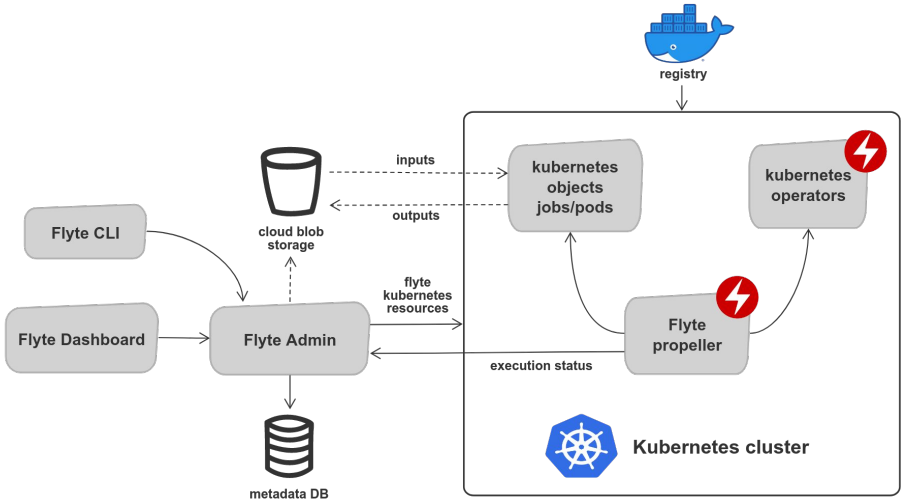


OPERATOR
FRAMEWORK
operatorframework.io

Flyte


registry
inputs
outputs
cloud blob storage
kubernetes objects jobs/pods
kubernetes operators
Flyte CLI
Flyte Dashboard
Flyte Admin
flyte kubernetes resources
Flyte propeller
execution status
metadata DB
Kubernetes cluster

Flyte



- Task execution and resource isolation is managed by Kubernetes
- Throttling and queueing is handled by Flyte propeller
- Multi tenant
- Allows to have isolated environment per task or project
- Supports workflow versioning
- Data aware, can cache task results

Overhead

- Ephemeral infrastructure brings a startup time overhead
- Teams needs to support their docker images

Anti patterns

- Table sensing is done much more elegant in Airflow (use event-driven approach)
- Not suited for a complex parallel computation

Good tool if you need a **multi tenant** environment, **custom dependencies** per task or project, **workflow versioning**, and **compute isolation** is required

- Good for classic ETL jobs
- Quick and simple to start
- Good support for table sensing
- No multi tenancy
- No workflow versioning
- Monolithic, fixed set of workers
- No infrastructure and environment isolation

- Good for a custom jobs (like ML)
- Multi tenant, api-friendly
- Supports workflow versioning
- Overhead with ephemeral infrastructure and image maintenance
- Infrastructure and environment isolation based on Kubernetes

Apache Airflow

Flyte

Choose the right tool for the right job*

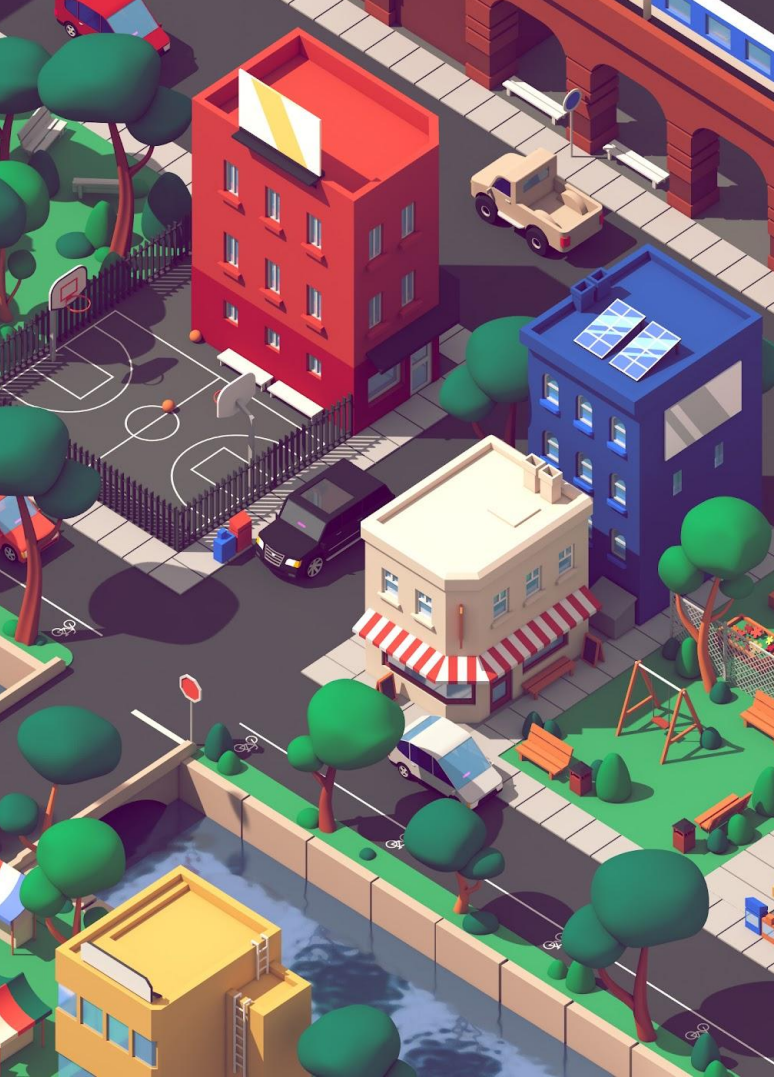
(Some cases can be implemented well on both engines)

Flyte

live demo

Please ask questions in the chat!
The best question gets something special from our speakers

Raffle Time!



Join the Ride!

Now Hiring in Minsk and Kyiv!

Backend, Data, ML Engineers

And many more on our careers page! (lyft.com/careers)

Connect with us!
LYFT.COM/CAREERS