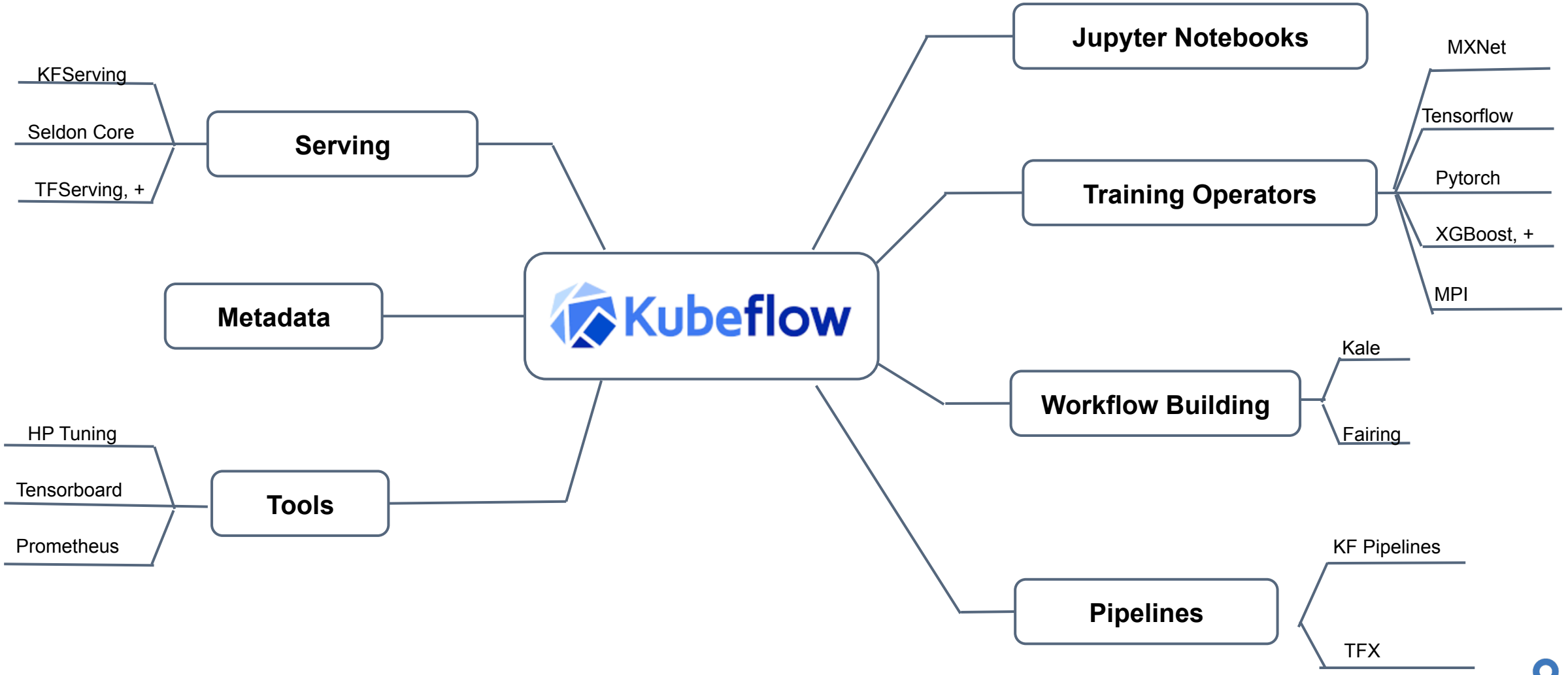
Kubeflow: End to End ML Platform

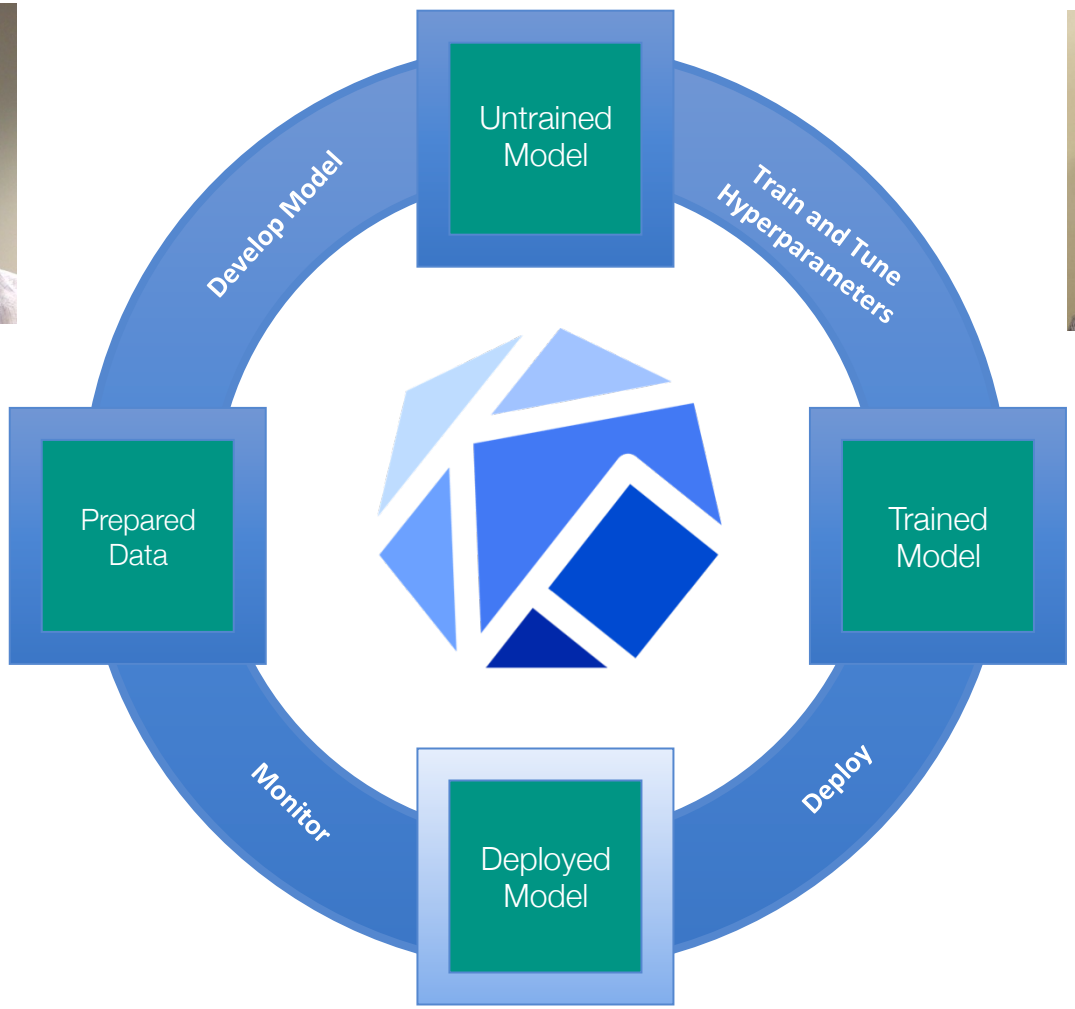
Animesh Singh

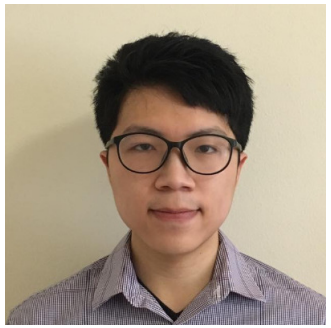


Kubeflow: IBM Contributors

Christian Kadner

Weiqiang Huang





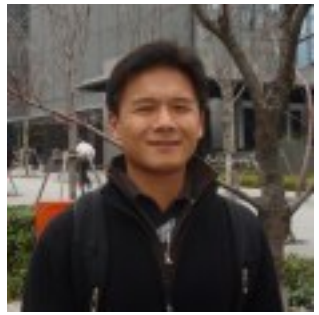
Tommy Li

Andrew Butler

Jin Chi He

Feng Li

Ke Zhu



Kevin Yu

Upstream, Midstream and Downstream

'Upstream' is about extracting oil and natural gas from the ground; 'midstream' is about safely moving them thousands of miles; and 'downstream' is converting these resources into the fuels and finished products we all depend on.

Upstream

Upstream has many phases, beginning with the exploratory process. Geologists search on dry land or in oceans for signs of gas or oil.

Midstream

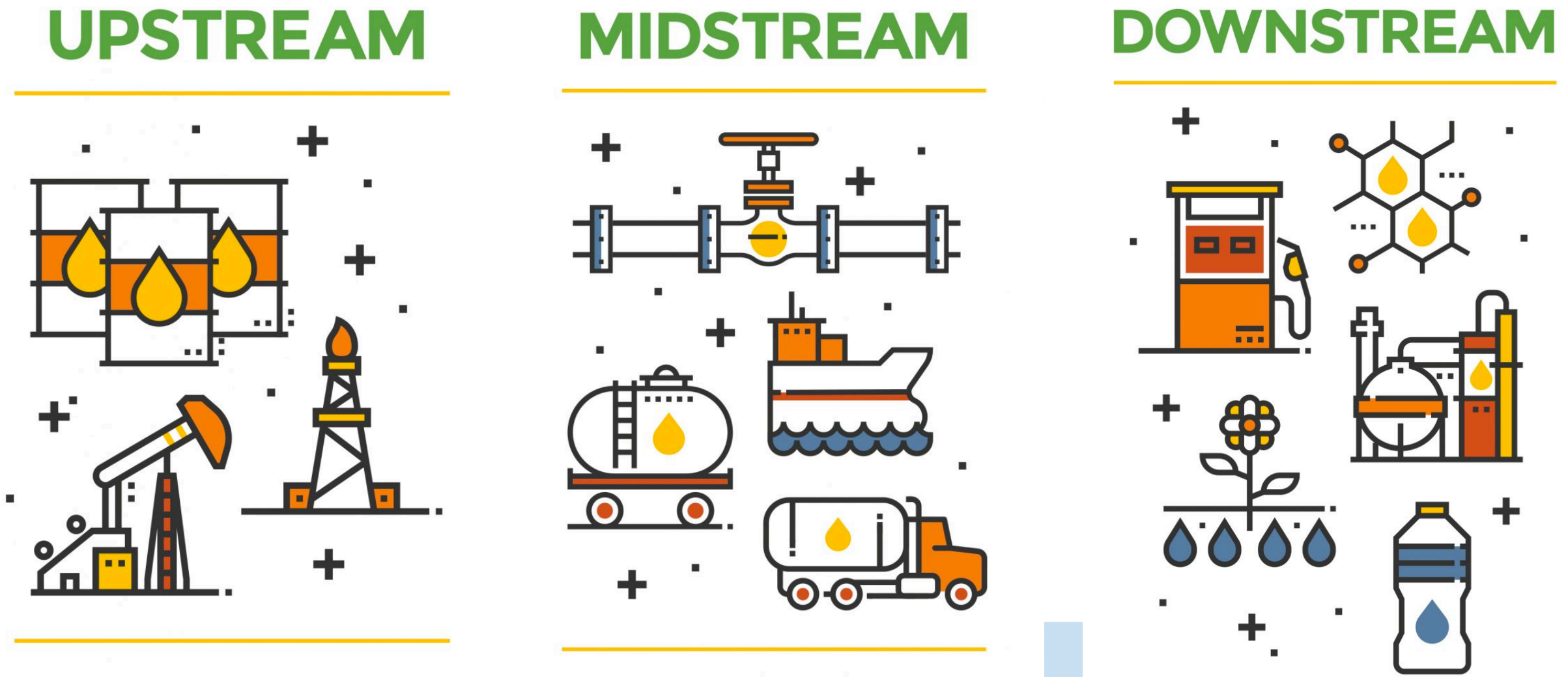
When a well is producing, oil or gas enters the midstream juncture.The middle part of the process requires multiple cooperation.

Downstream

The downstream stage handles processing, selling, marketing and distributing gas or oil. Final products depend upon the initial resource.

Upstream, Midstream and Downstream

'Upstream' is about extracting oil and natural gas from the ground; 'midstream' is about safely moving them thousands of miles; and 'downstream' is converting these resources into the fuels and finished products we all depend on.



Kubeflow

https://github.com/kubeflow

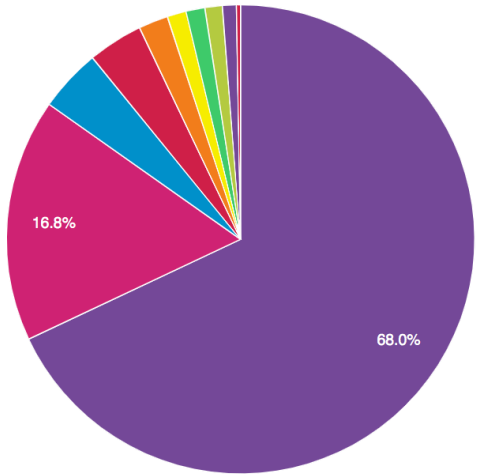
- End to end ML Platform on Kubernetes. Focused on multiple aspects of Model Lifecycle
- Originated at Google, and has grown to have a large community of developers
- Google, IBM, Cisco, RedHat, Intel, Microsoft, AWS and others contributing
- IBM is the 2nd contributor in terms of overall commits. IBM maintainers (committers/reviewers) in Katib (HPO+Training), Kubeflow Serving, Manifests, Pipelines etc.

Commits by Company

Show 10 entries Search

#	Company	Commits
	*independent	3669
1	Google	904
2	IBM	234
3	Caicloud	205
4	Alibaba	110
5	Red Hat	71
6	Intel	65
7	Huawei	52
8	Cisco Systems	15
9	VA Linux	12

Showing 1 to 10 of 29 entries Previous Next



Libraries and CLIs - Focus on end users

Arena | kfctl | kubectl | fairing

Systems - Combine multiple services

katib | pipelines | Model DB | kube bench | notebooks | TFX

Low Level APIs / Services (single function)

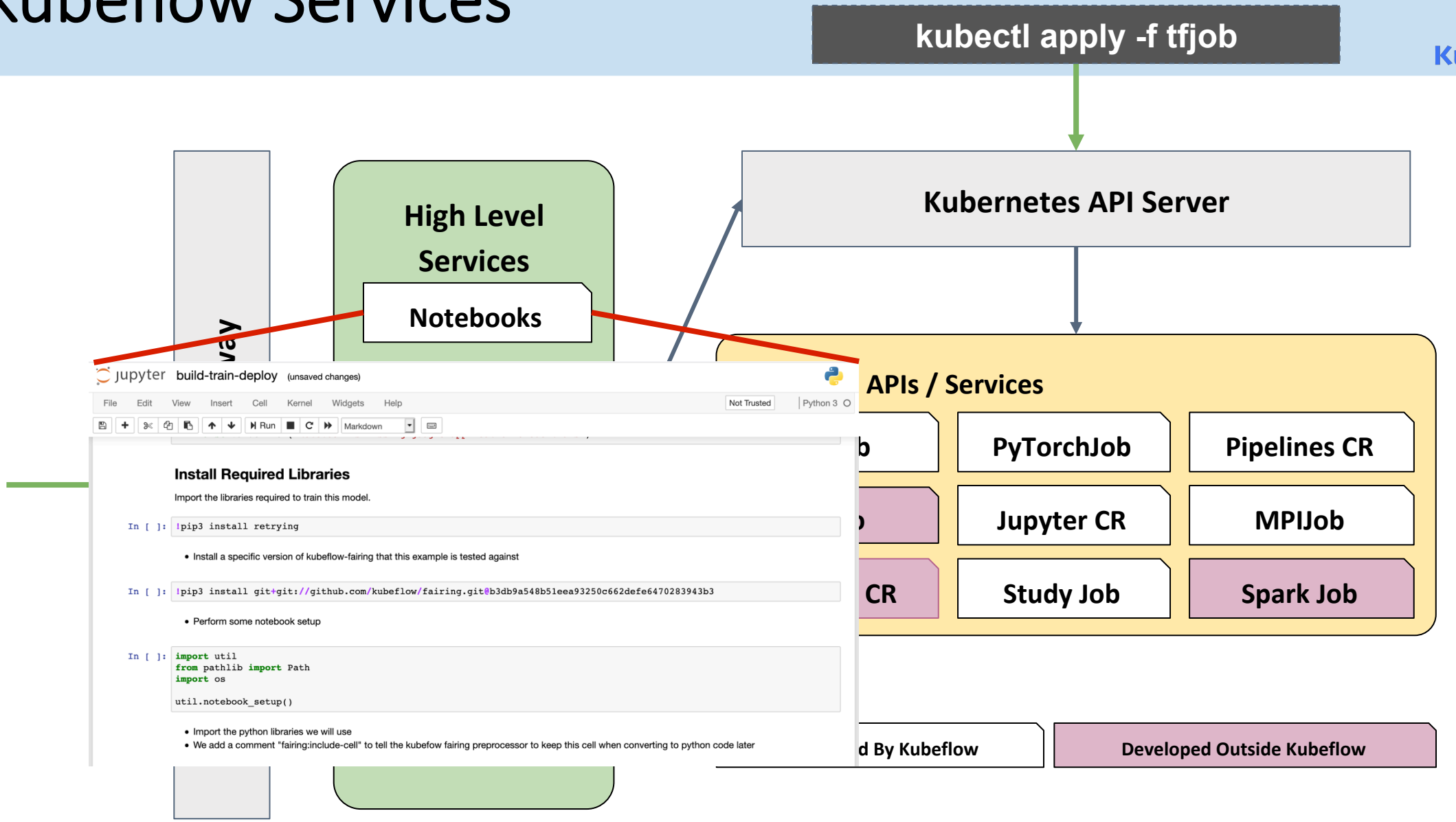
TFJob | PyTorchJob | Pipelines CR | Argo | Jupyter CR | MPI CR | Seldon CR | Study Job | Spark Job

Metadata | IAM | Orchestration | Scheduling

Developed By Kubeflow | Developed Outside Kubeflow

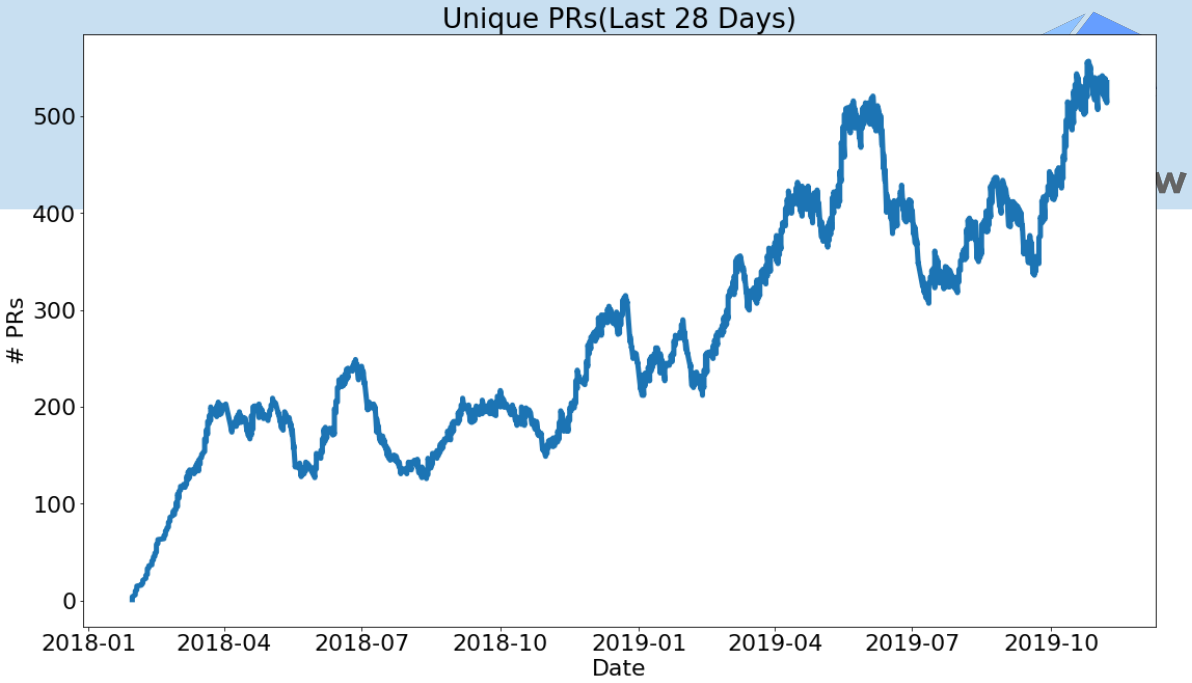
* Not all components shown

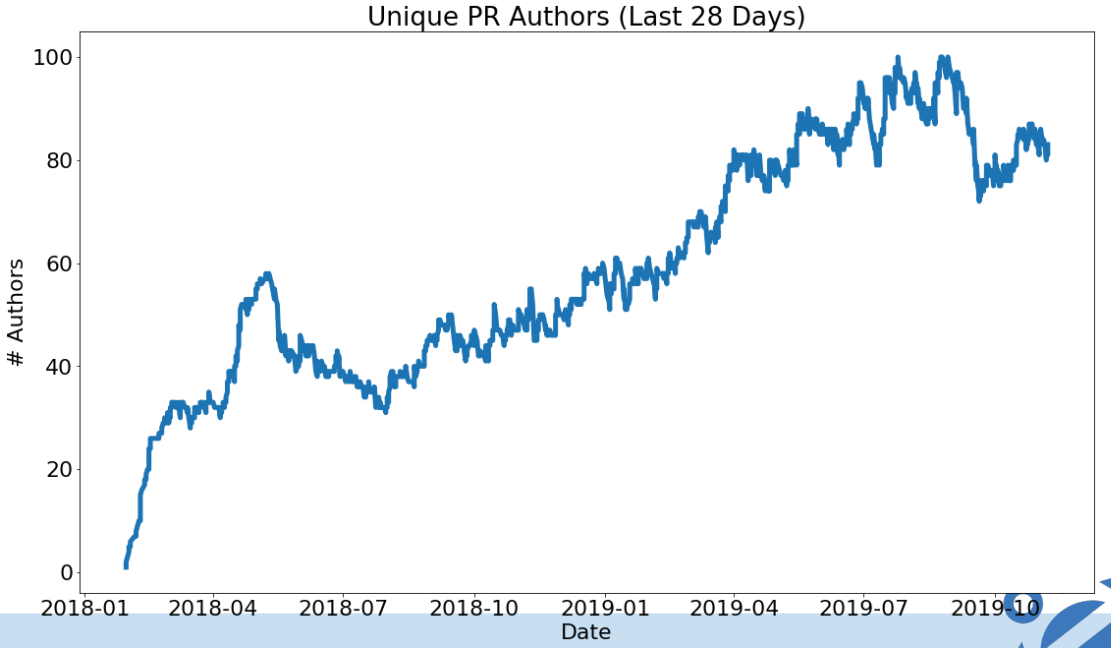
Kubeflow Services



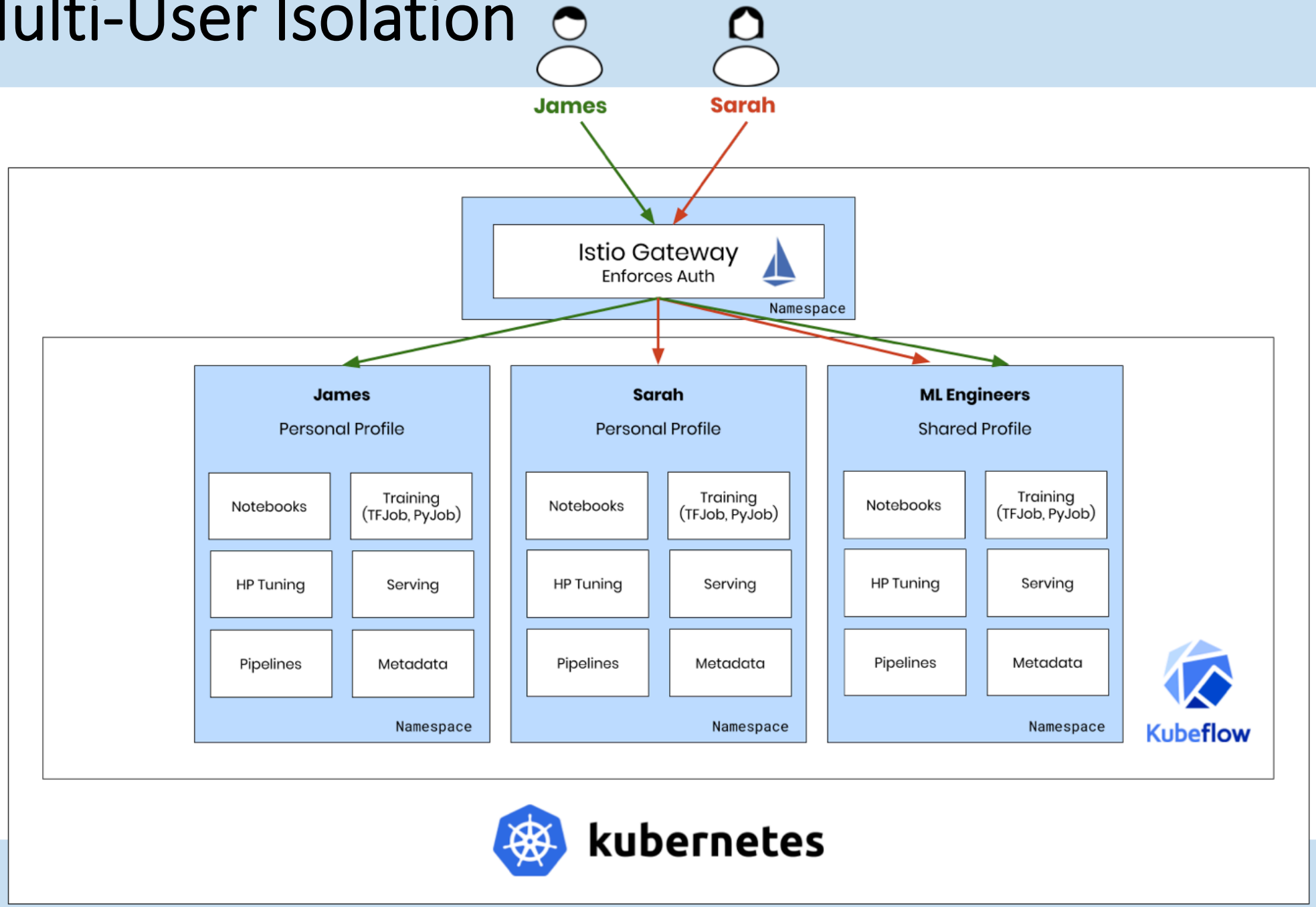
Adapted from Kubeflow Contributor Summit 2019 talk: Kubeflow and ML Landscape (Not all components are shown)

Community is growing!

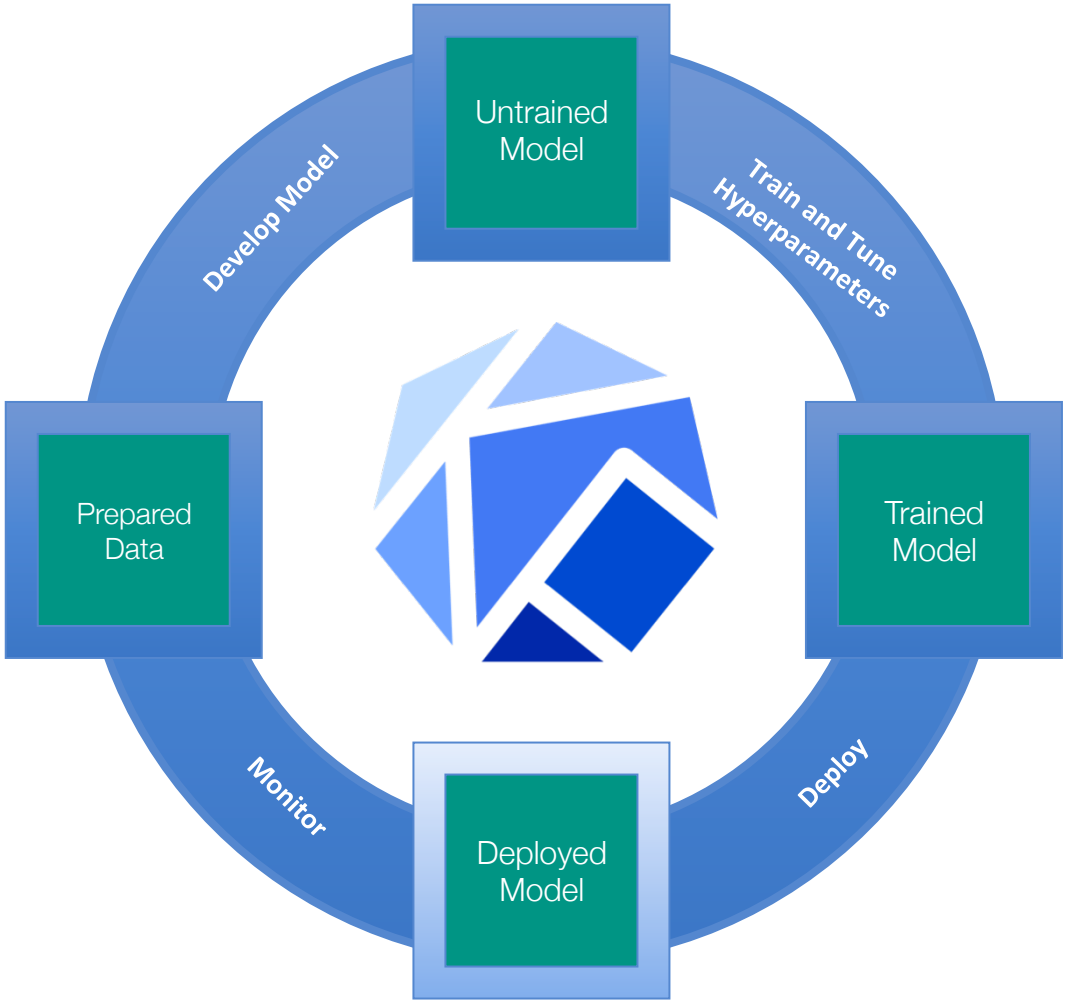




Multi-User Isolation



ML Lifecycle: Build: Development, Training and HPO



Develop (Kubeflow Jupyter Notebooks)

- Data Scientist
- Self-service Jupyter Notebooks provide faster model experimentation
- Simplified configuration of CPU/GPU, RAM, Persistent Volumes
- Faster model creation with training operators, TFX, magics, workflow automation (Kale, Fairing)
- Simplify access to external data sources (using stored secrets)
- Easier protection, faster restoration & sharing of “complete” notebooks

- IT Operator
- Profile Controller, Istio, Dex enable secure RBAC to notebooks, data & resources
- Smaller base container images for notebooks, fewer crashes, faster to recover

Distributed Training Operators

	TF Operator	PyTorch Operator	MPI Operator
Framework Support	TensorFlow	PyTorch	HOROVOD TensorFlow/Keras Apache MXNet/PyTorch/OpenMPI
Distribution Strategy & Backend	tf.distribute MPI/NCCL/PS/TPU	torch.distributed Gloo/MPI/NCCL	horovod DistributedOptimizer Gloo/MPI/NCCL

Distributed Training Operators

tf-operator
Tools for ML/Tensorflow on Kubernetes.
Jsonnet Apache-2.0 323

pytorch-operator
PyTorch on Kubernetes
Jsonnet Apache-2.0 87

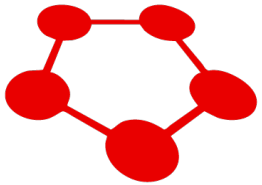
mpi-operator
Kubernetes Operator for Allreduce-style
kubernetes tensorflow mpi dist
horovod kubeflow
Go Apache-2.0 83 125

xgboost-operator
Incubating project for xgboost operator
Go Apache-2.0 23 41

mxnet-operator
A Kubernetes operator for mxnet jobs
Go Apache-2.0 20 50

chainer-operator
Repository for chainer operator
Go Apache-2.0 9 12

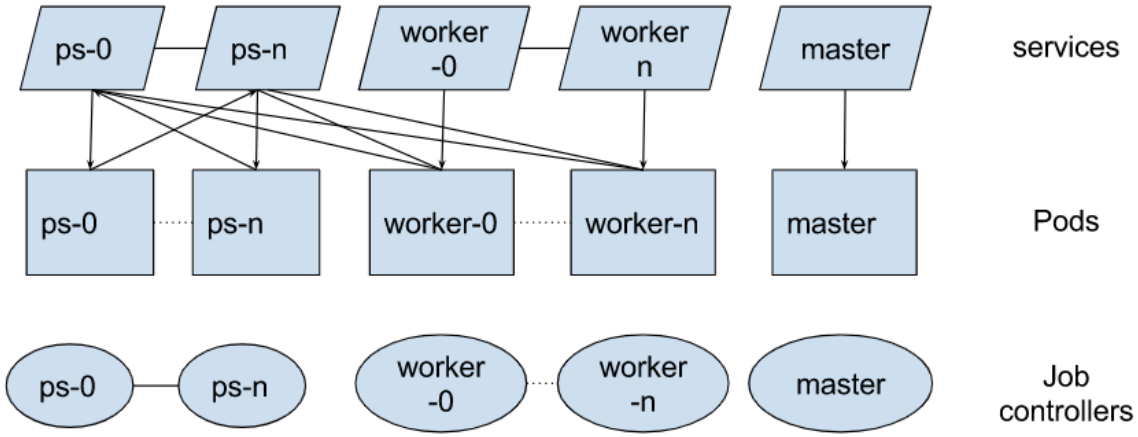
XGBoost



Chainer

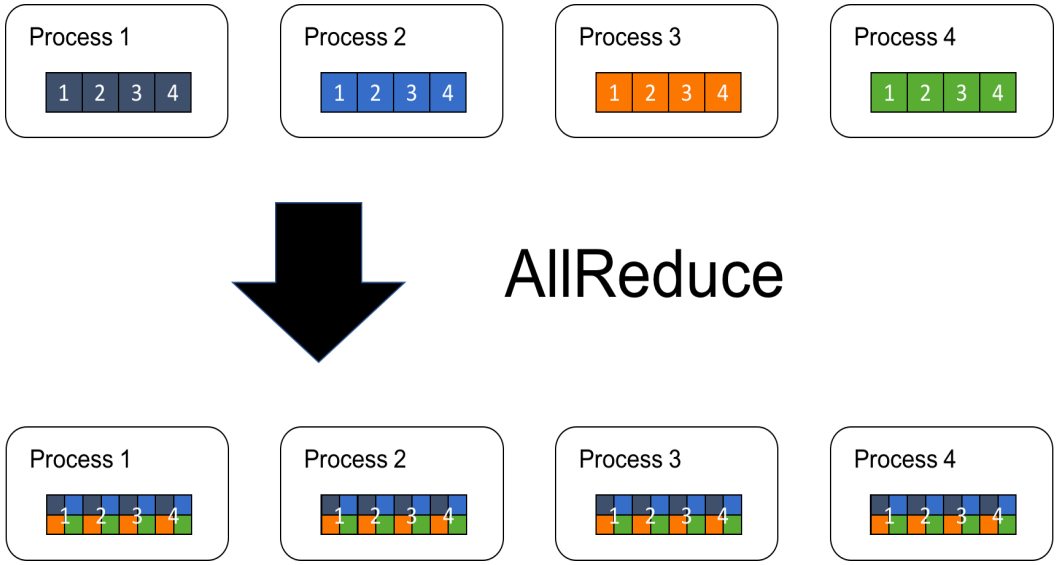
Distributed Tensorflow Operator

- A distributed Tensorflow Job is collection of the following processes
  - Chief – The chief is responsible for orchestrating training and performing tasks like checkpointing the model
  - Ps – The ps are parameters servers; the servers provide a distributed data store for the model parameters to access
  - Worker – The workers do the actual work of training the model. In some cases, worker 0 might also act as the chief
  - Evaluator - The evaluators can be used to compute evaluation metrics as the model is trained



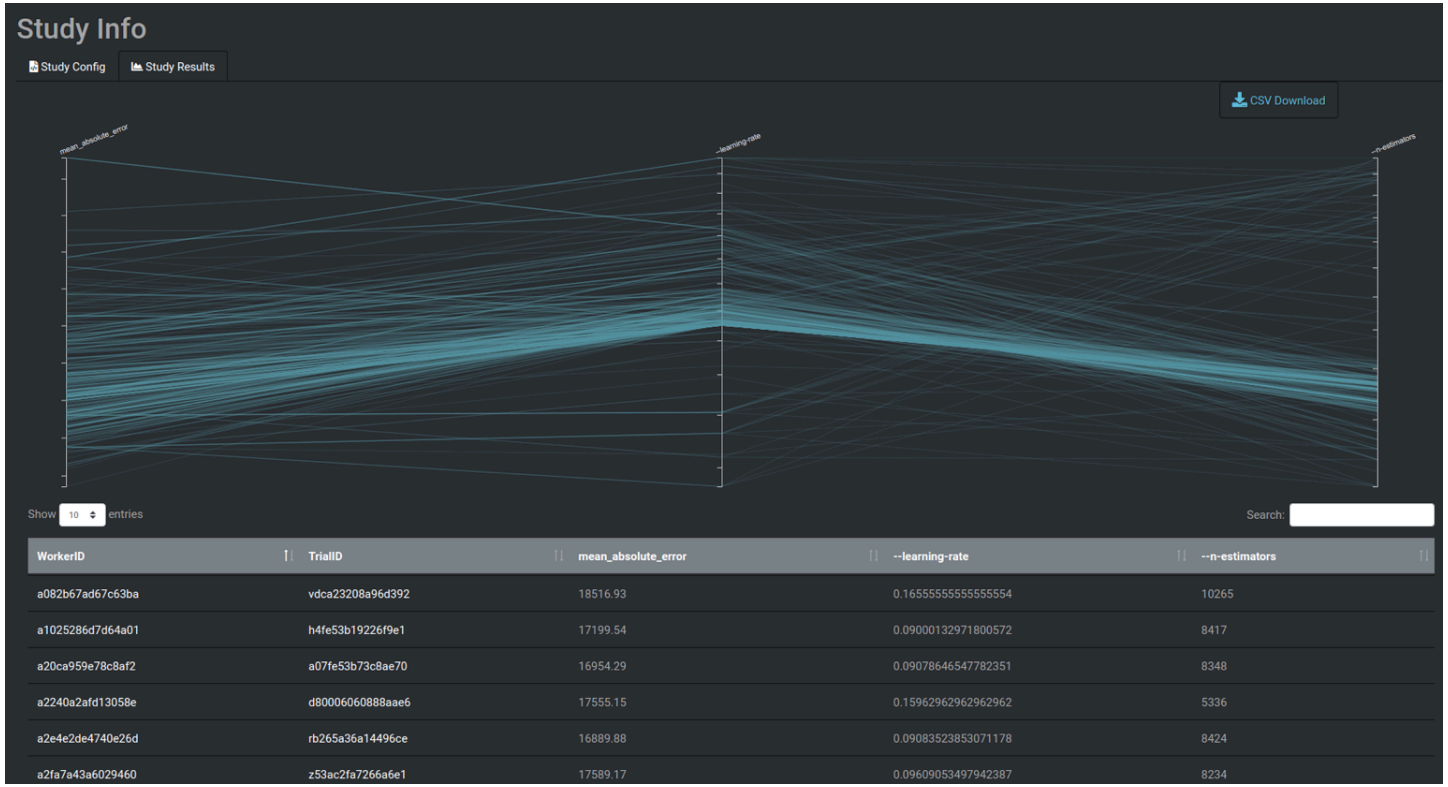
Distributed MPI Operator - AllReduce

- AllReduce is an operation that reduces many arrays spread across multiple processes into a single array which can be returned to all the processes
- This ensures consistency between distributed processes while allowing all of them to take on different workloads
- The operation used to reduce the multiple arrays back into a single array can vary and that is what makes the different options for AllReduce



Hyper Parameter Optimization and Neural Architecture Search - Katib

- Katib: Kubernetes Native System for Automated tuning of machine learning model's Hyperparameter Turning and Neural Architecture Search.
- Github Repository: https://github.com/kubeflow/katib



- Hyperparameter Tuning
  - ❑ Random Search
  - ❑ Tree of Parzen Estimators (TPE)
  - ❑ Grid Search
  - ❑ Hyperband
  - ❑ Bayesian Optimization
  - ❑ CMA Evolution Strategy
- Neural Architecture Search
  - ❑ Efficient Neural Architecture Search (ENAS)
  - ❑ Differentiable Architecture Search (DARTS)

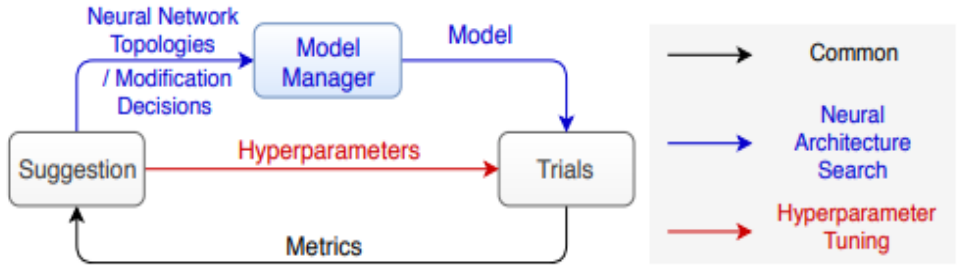
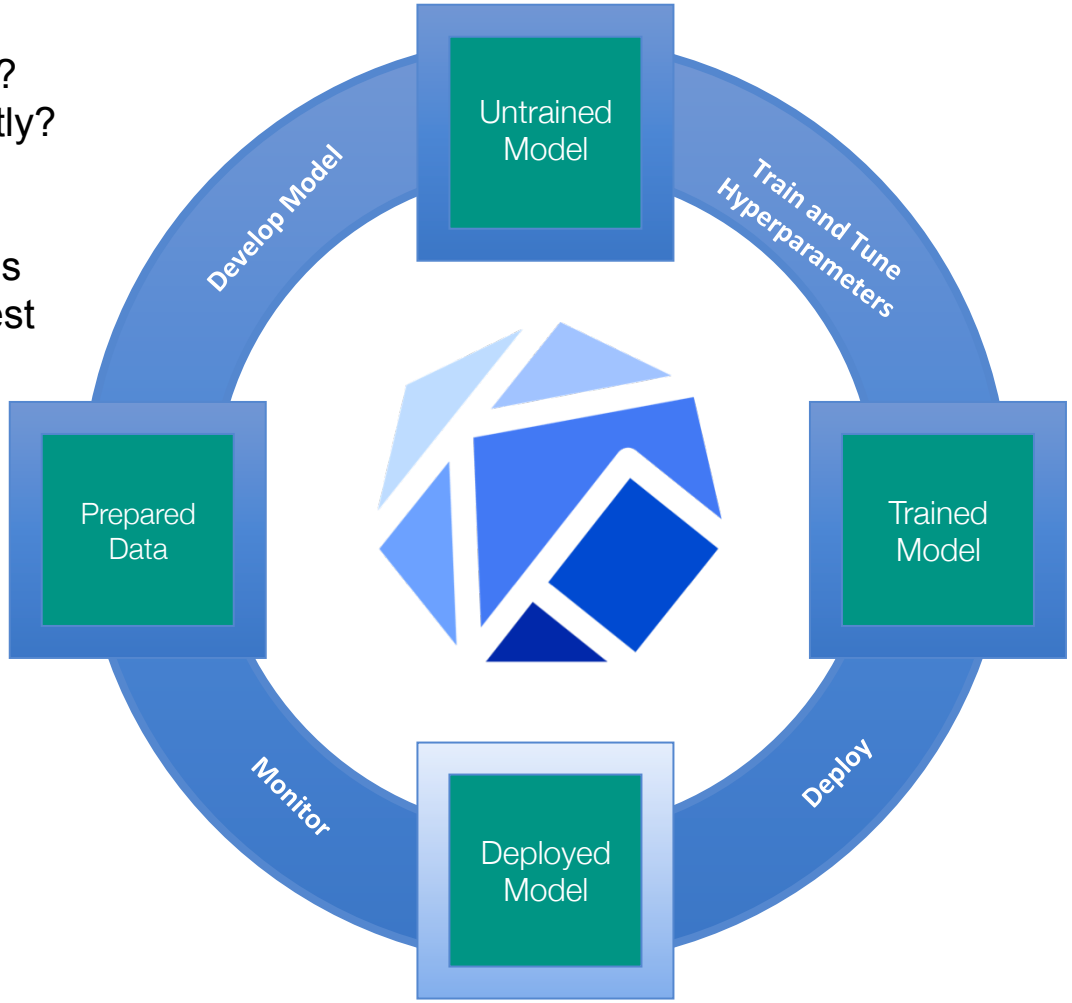


Figure 1: Summary of AutoML workflows

ML Lifecycle: Production Model Serving

- Cost:
  Is the model over or under scaled?
  Are resources being used efficiently?
- Monitoring:
  Are the endpoints healthy? What is the performance profile and request trace?
- Rollouts:
  Is this rollout safe? How do I roll back? Can I test a change without swapping traffic?
- Protocol Standards:
  How do I make a prediction? GRPC? HTTP? Kafka?



- How do I handle batch predictions?
- How do I leverage standardized Data Plane protocol so that I can move my model across MLServing platforms?
- Frameworks:
  How do I serve on Tensorflow? XGBoost? Scikit Learn? Pytorch? Custom Code?
- Features:
  How do I explain the predictions? What about detecting outliers and skew? Bias detection? Adversarial Detection?
- How do I wire up custom pre and post processing

Experts fragmented across industry

- Seldon Core was pioneering Graph Inferencing.
- IBM and Bloomberg were exploring serverless ML lambdas. IBM gave a talk on the ML Serving with Knative at last KubeCon in Seattle
- Google had built a common Tensorflow HTTP API for models.
- Microsoft Kubernetizing their Azure ML Stack

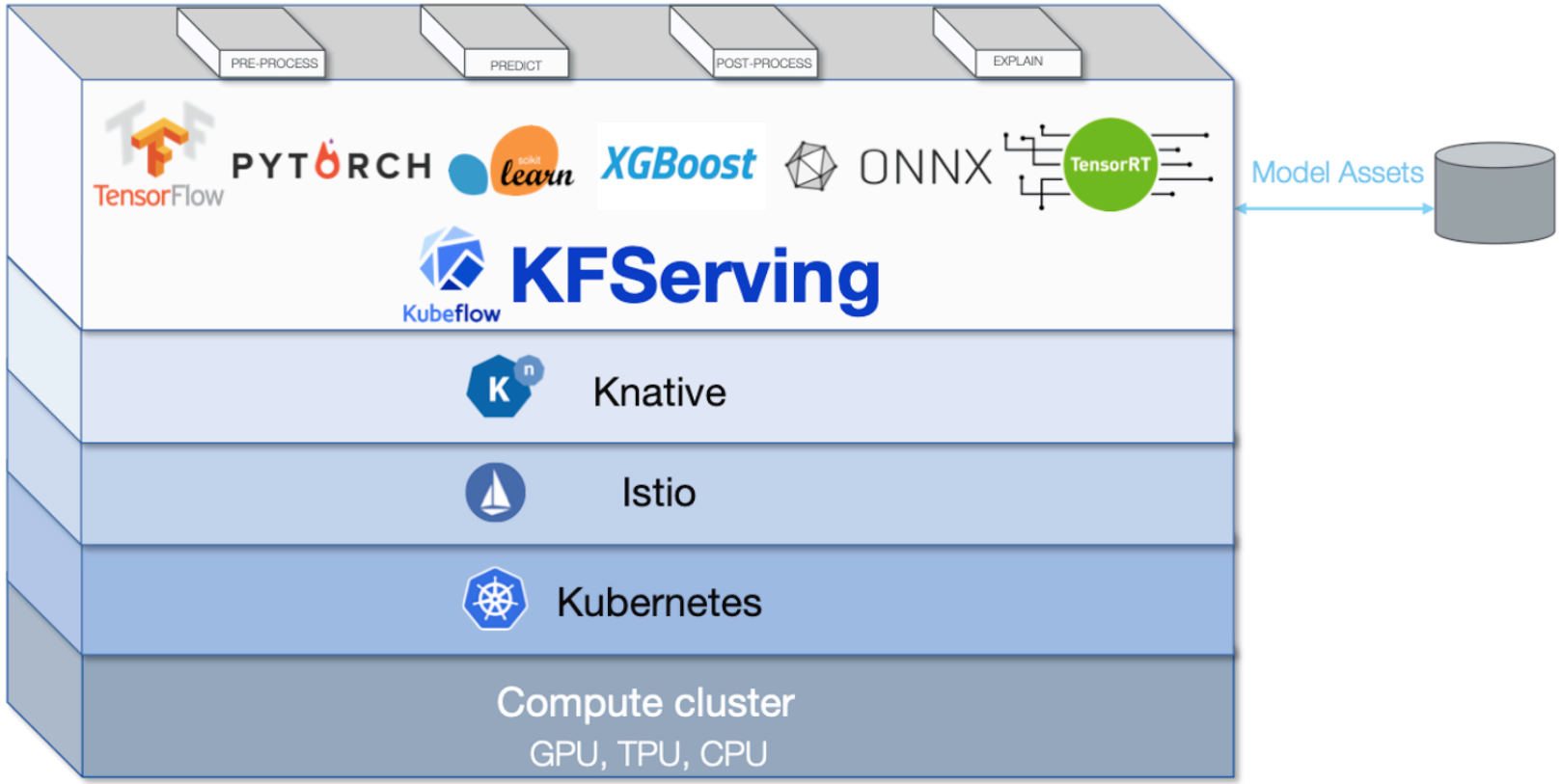
Putting the pieces together

- Kubeflow created the conditions for collaboration.
- A promise of open code and open community.
- Shared responsibilities and expertise across multiple companies.
- Diverse requirements from different customer segments

Google SELDON IBM Bloomberg Microsoft

Model Serving - KFServing

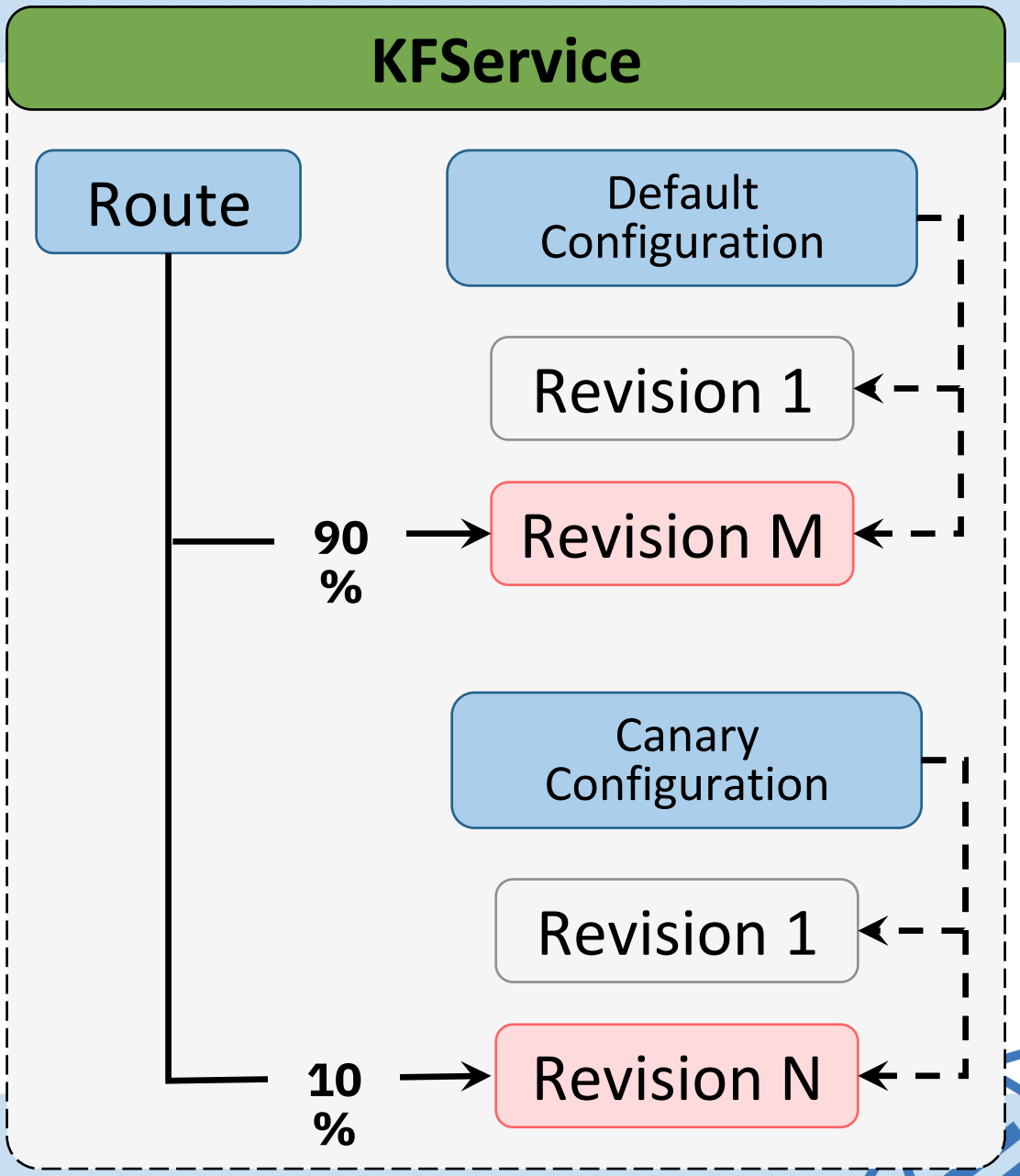
- Founded by Google, Seldon, IBM, Bloomberg and Microsoft
- Part of the Kubeflow project
- Focus on 80% use cases - single model rollout and update
- Kfserving 1.0 goals:
  - Serverless ML Inference
  - Canary rollouts
  - Model Explanations
  - Optional Pre/Post processing



KFServing: Default and Canary Configurations

Manages the hosting aspects of your models

- **InferenceService** - manages the lifecycle of models
- **Configuration** - manages history of model deployments. Two configurations for default and canary.
- **Revision** - A snapshot of your model version
- **Route** - Endpoint and network traffic management



Supported Frameworks, Components and Storage Subsystems

Model Servers

- TensorFlow
- Nvidia TRTIS
- PyTorch
- XGBoost
- SKLearn
- ONNX

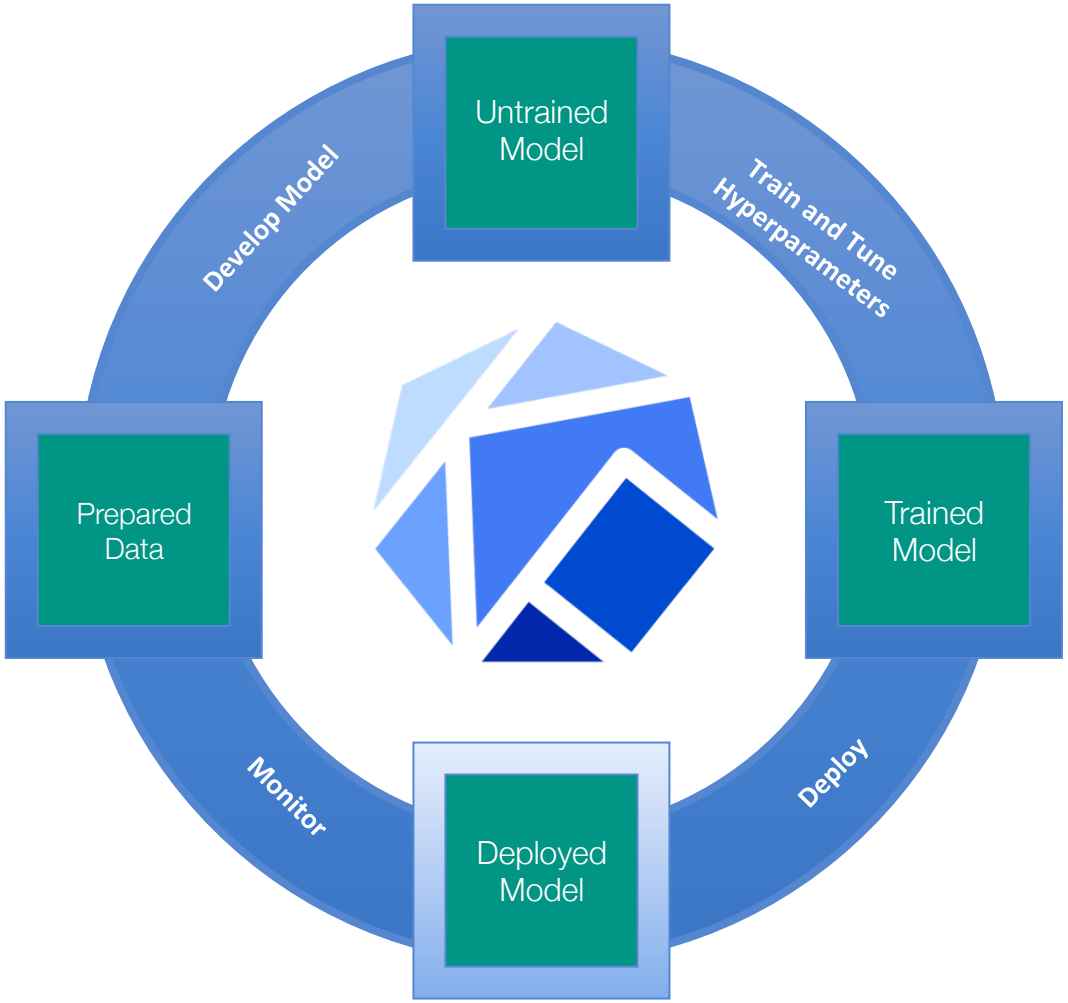
Components:

- - Predictor, Explainer, Transformer (pre-processor, post-processor)

Storage

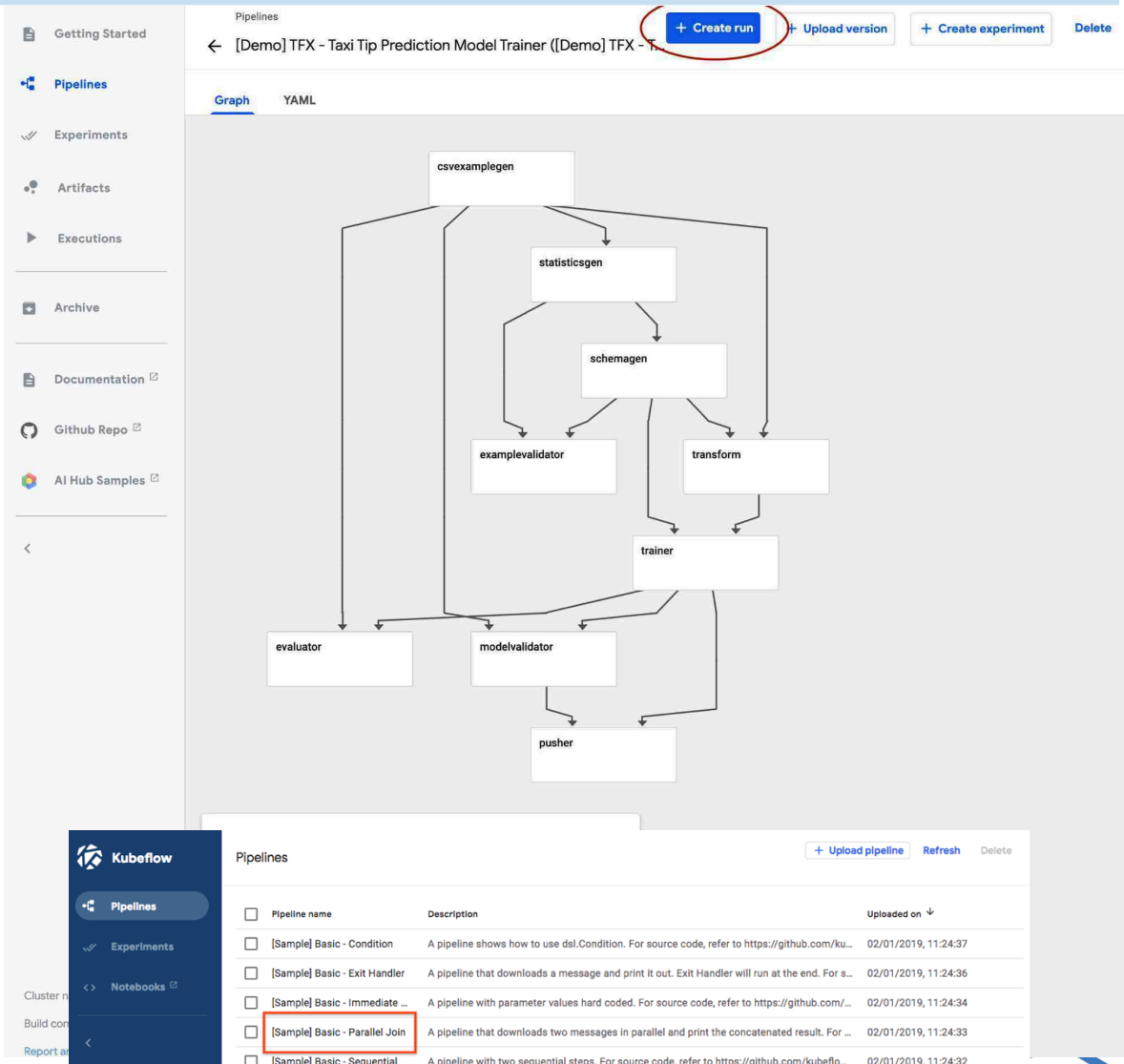
- AWS/S3
- GCS
- Azure Blob
- PVC

ML Lifecycle: Orchestrate Build, Train, Validate and Deploy

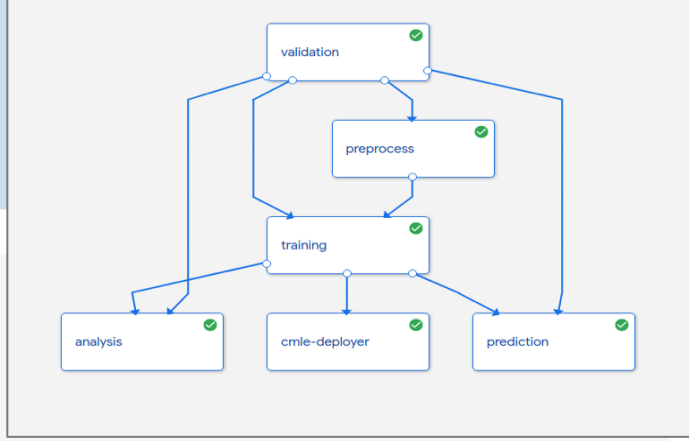


Kubeflow Pipelines

- Containerized implementations of ML Tasks
  - Pre-built components: Just provide params or code snippets (e.g. training code)
  - Create your own components from code or libraries
  - Use any runtime, framework, data types
  - Attach k8s objects - volumes, secrets
- Specification of the sequence of steps
  - Specified via Python DSL
  - Inferred from data dependencies on input/output
- Input Parameters
  - A “Run” = Pipeline invoked w/ specific parameters
  - Can be cloned with different parameters
- Schedules
  - Invoke a single run or create a recurring scheduled pipeline



Define Pipeline with Python SDK



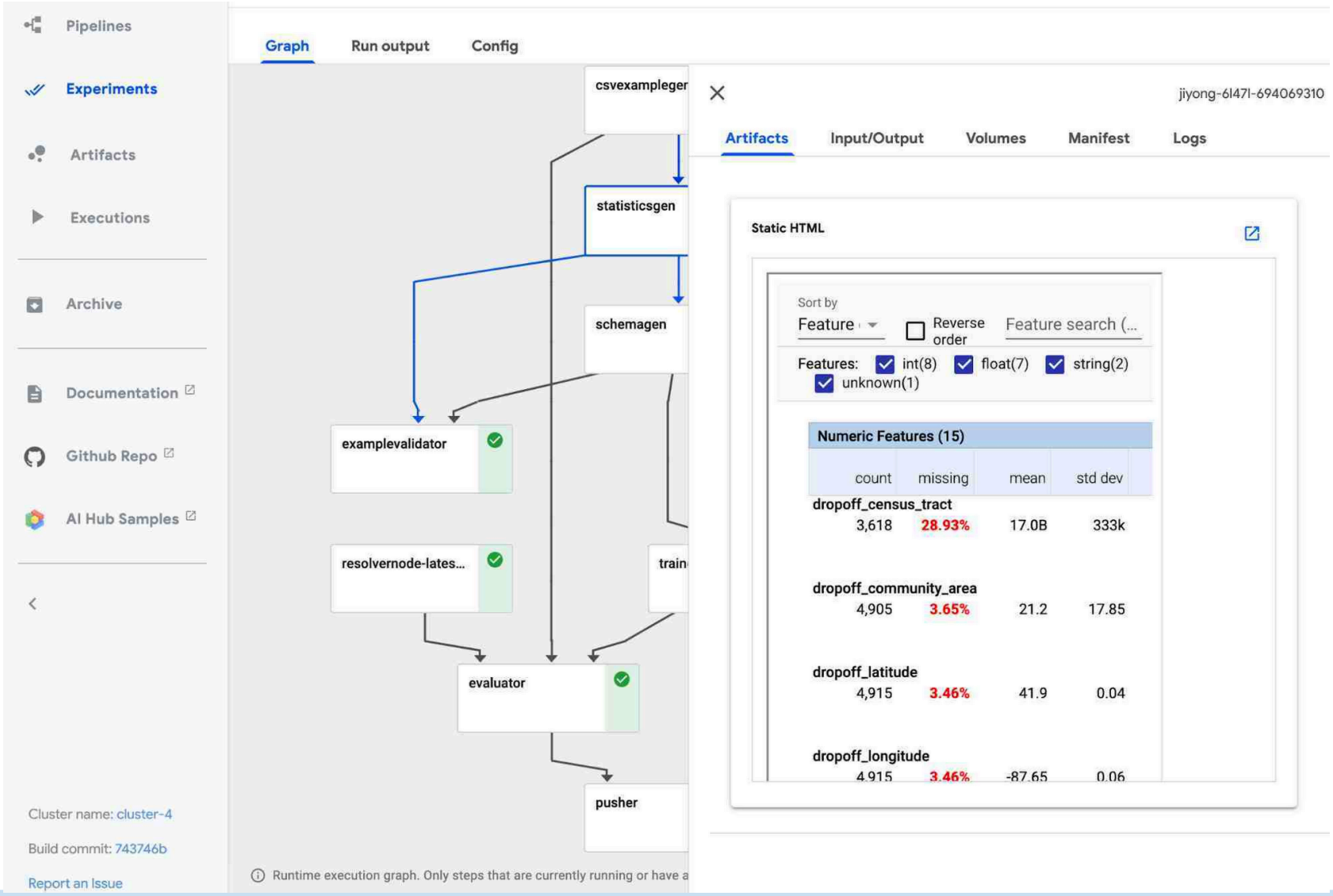

```
@dsl.pipeline(name='Taxi Cab Classification Pipeline Example’)
def taxi_cab_classification(
    output_dir,
    project,
    Train_data      = 'gs://bucket/train.csv',
    Evaluation_data = 'gs://bucket/eval.csv',
    Target          = 'tips',
    Learning_rate   = 0.1, hidden_layer_size = '100,50’, steps=3000):

        tfdv                = TfdvOp(train_data, evaluation_data, project, output_dir)
        preprocess          = PreprocessOp(train_data, evaluation_data, tfdv.output[“schema”], project, output_dir)
        training  = DnnTrainerOp(preprocess.output, tfdv.schema, learning_rate, hidden_layer_size, steps,
                                 target, output_dir)
        tfma                = TfmaOp(training.output, evaluation_data, tfdv.schema, project, output_dir)
        deploy    = TfServingDeployerOp(training.output)
```

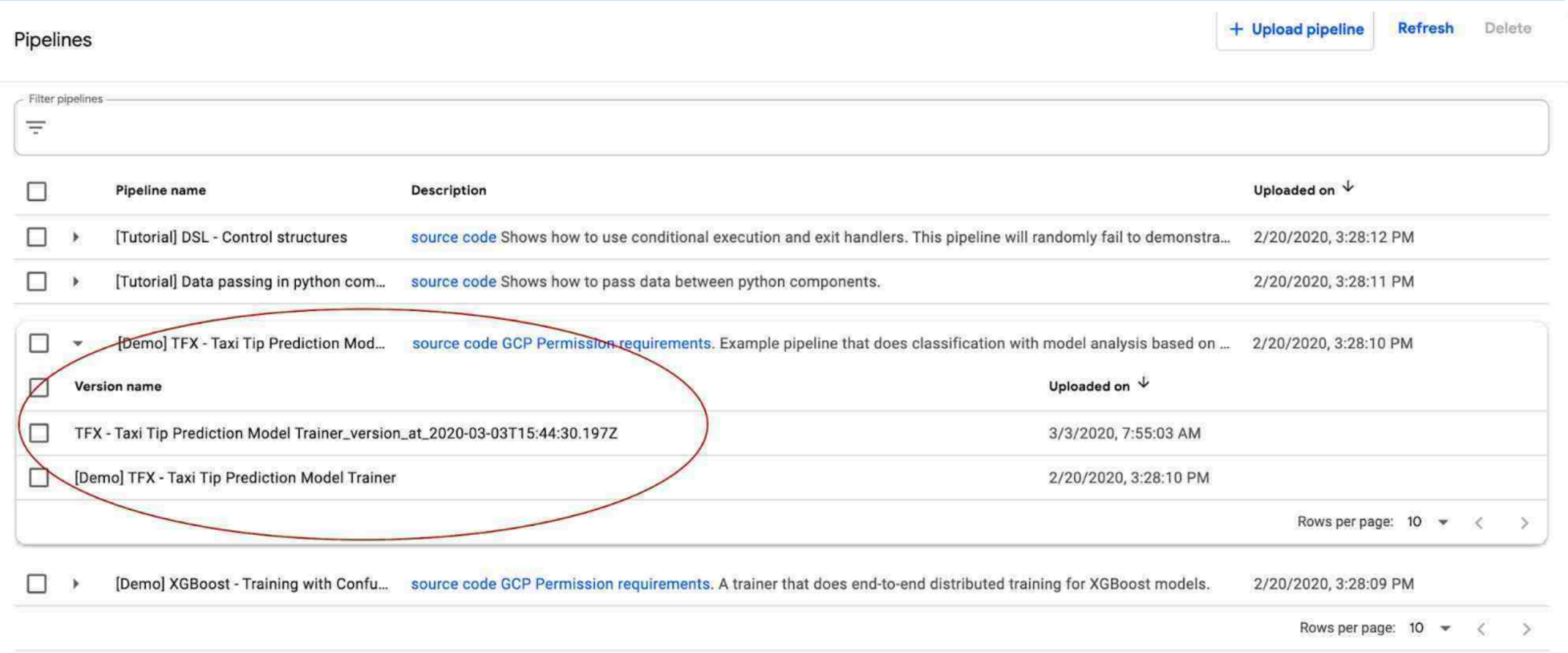
Compile and Submit Pipeline Run

```
dsl.compile(taxi_cab_classification,  'tfx.tar.gz')
run = client.run_pipeline(
      'tfx_run', 'tfx.tar.gz', params={'output': ‘gs://dpa22’, 'project': ‘my-project-33’})
```

Visualize the state of various components

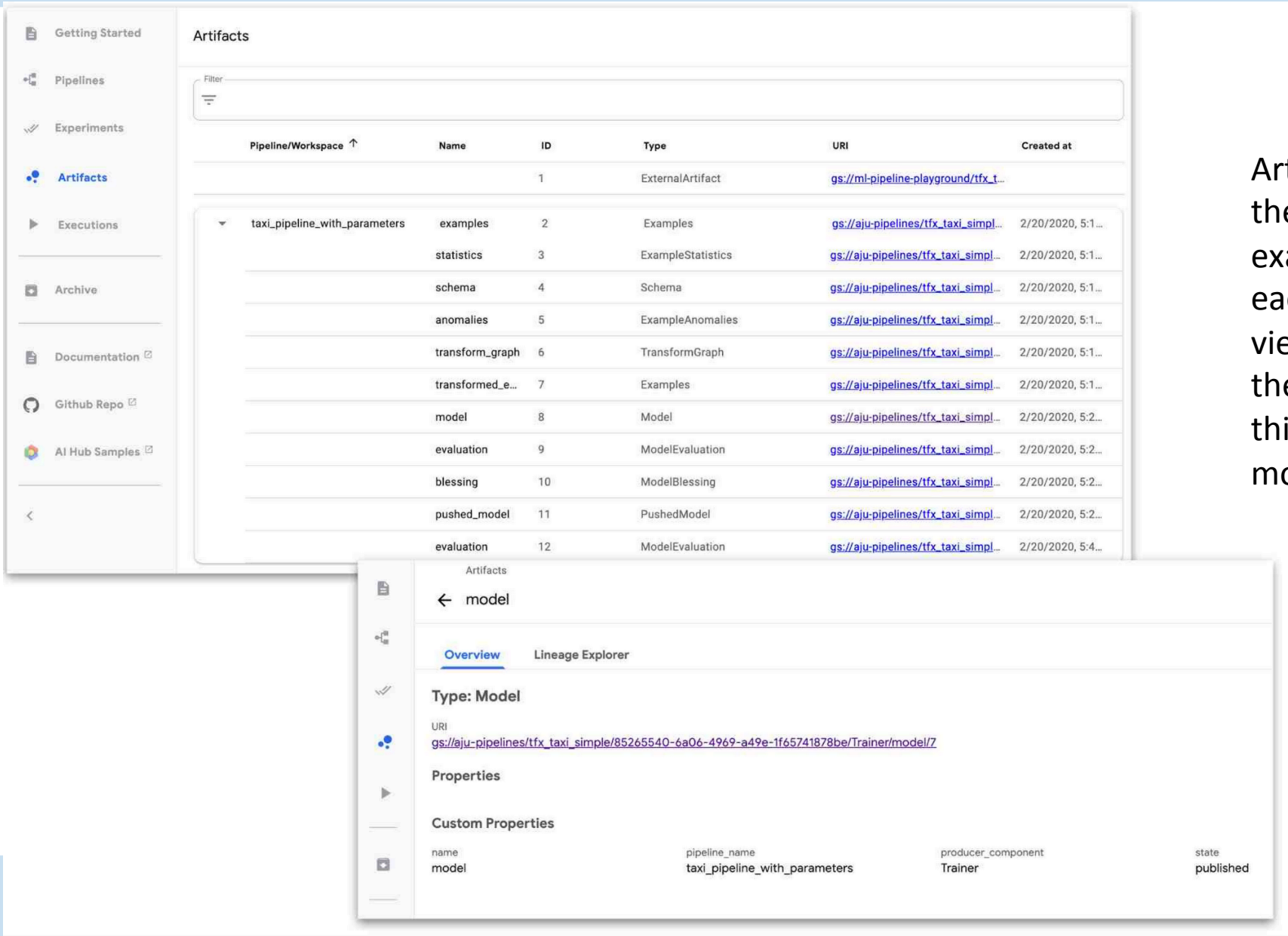


Pipelines versioning



Pipelines lets you group and manage multiple versions of a pipeline.

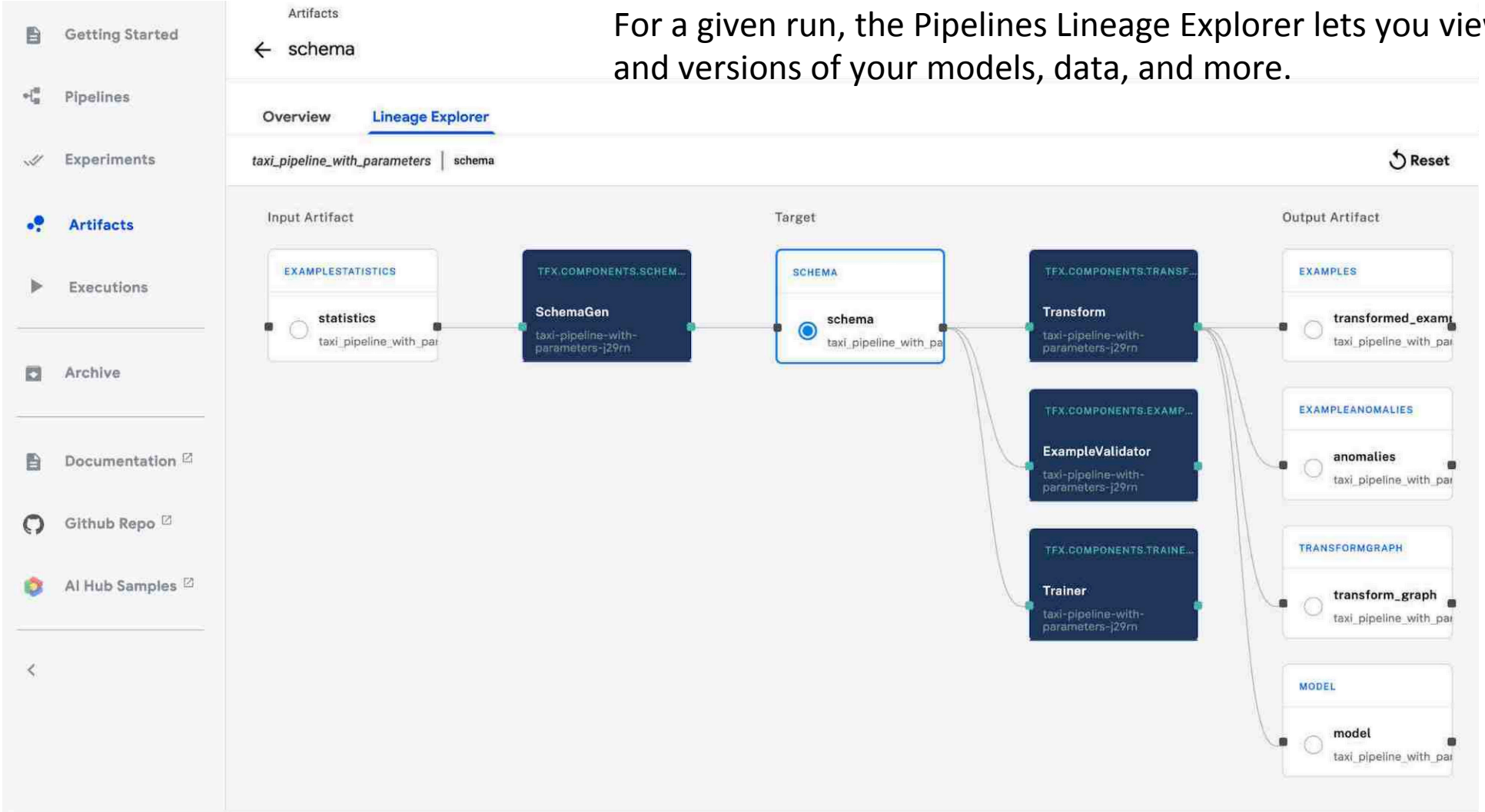
Artifact Tracking



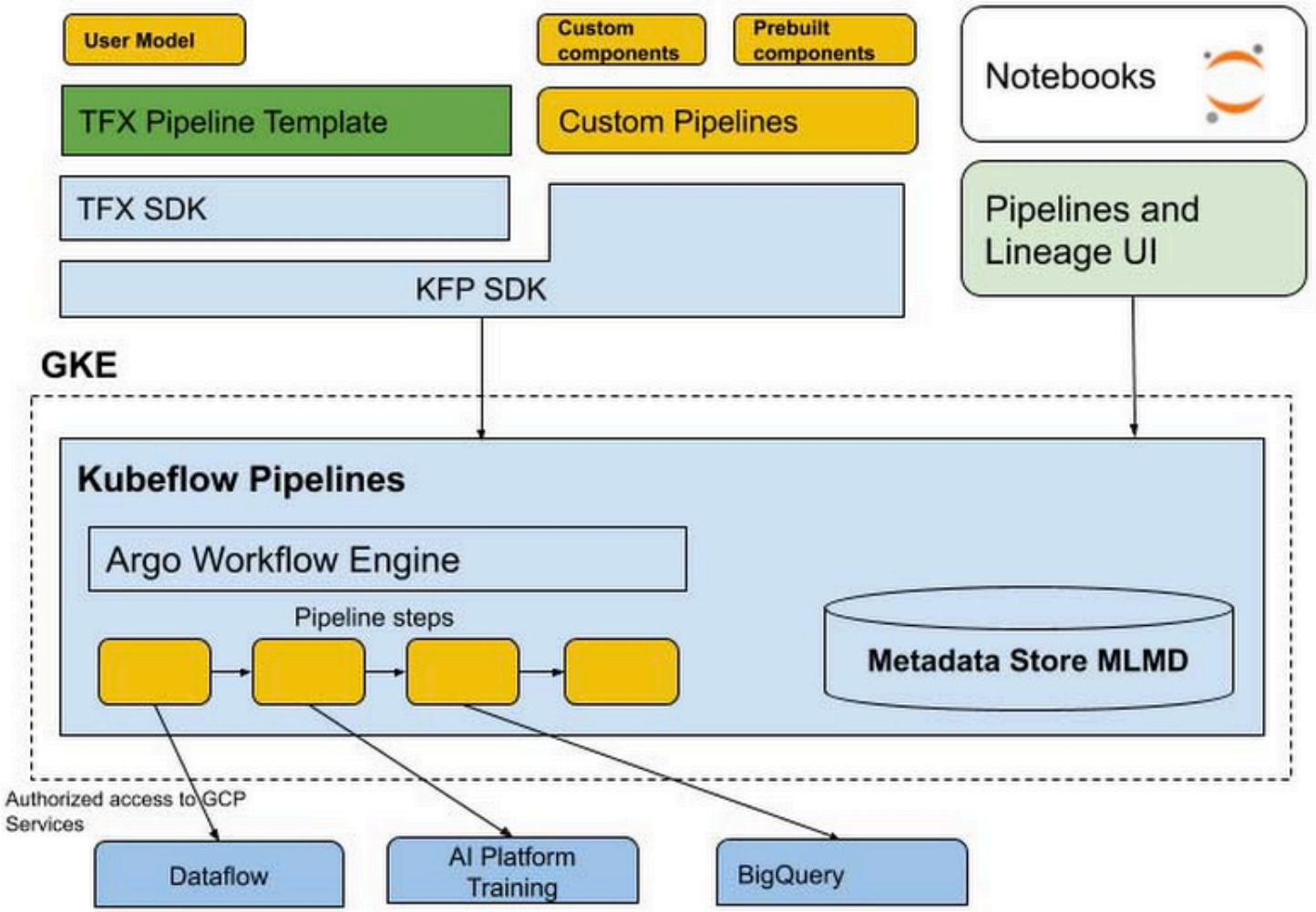
Artifacts for a run of the “TFX Taxi Trip” example pipeline. For each artifact, you can view details and get the artifact URL—in this case, for the model.

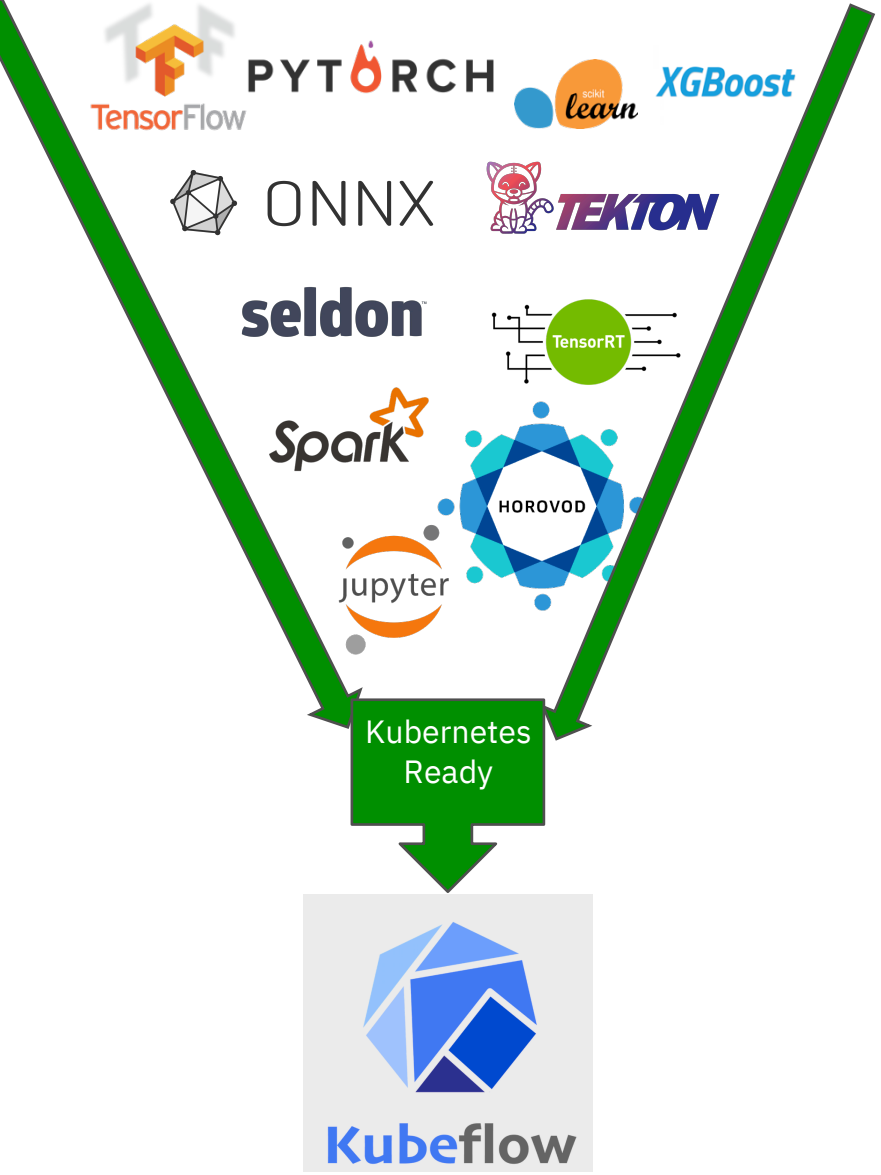
Lineage Tracking

For a given run, the Pipelines Lineage Explorer lets you view the history and versions of your models, data, and more.



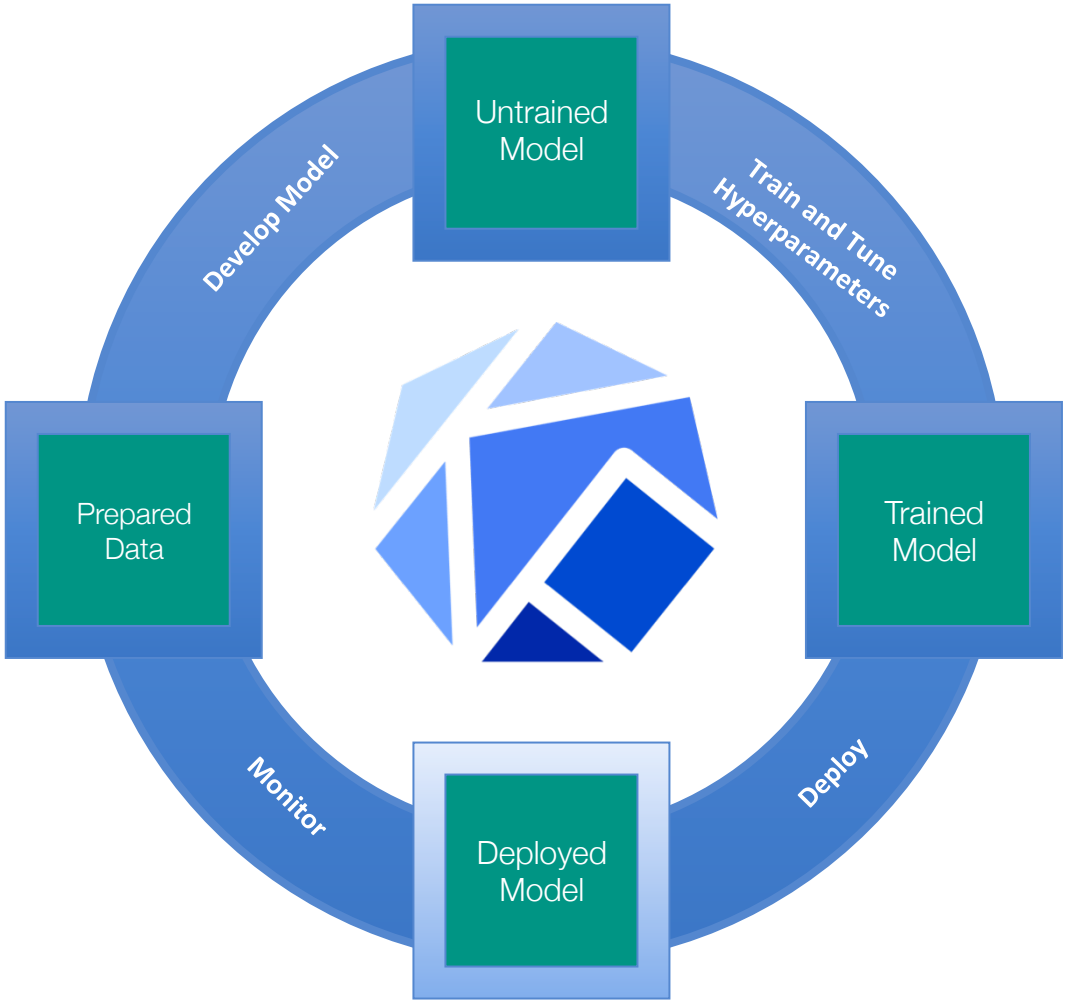
Kubeflow Pipeline Architecture





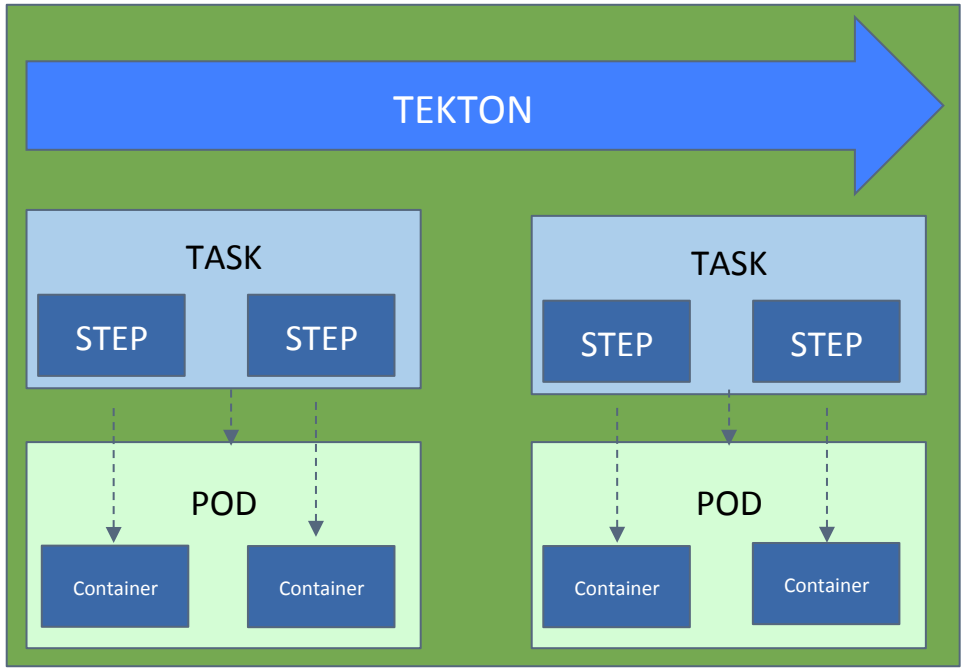
ML and AI Platform

Kubeflow Pipelines with Tekton



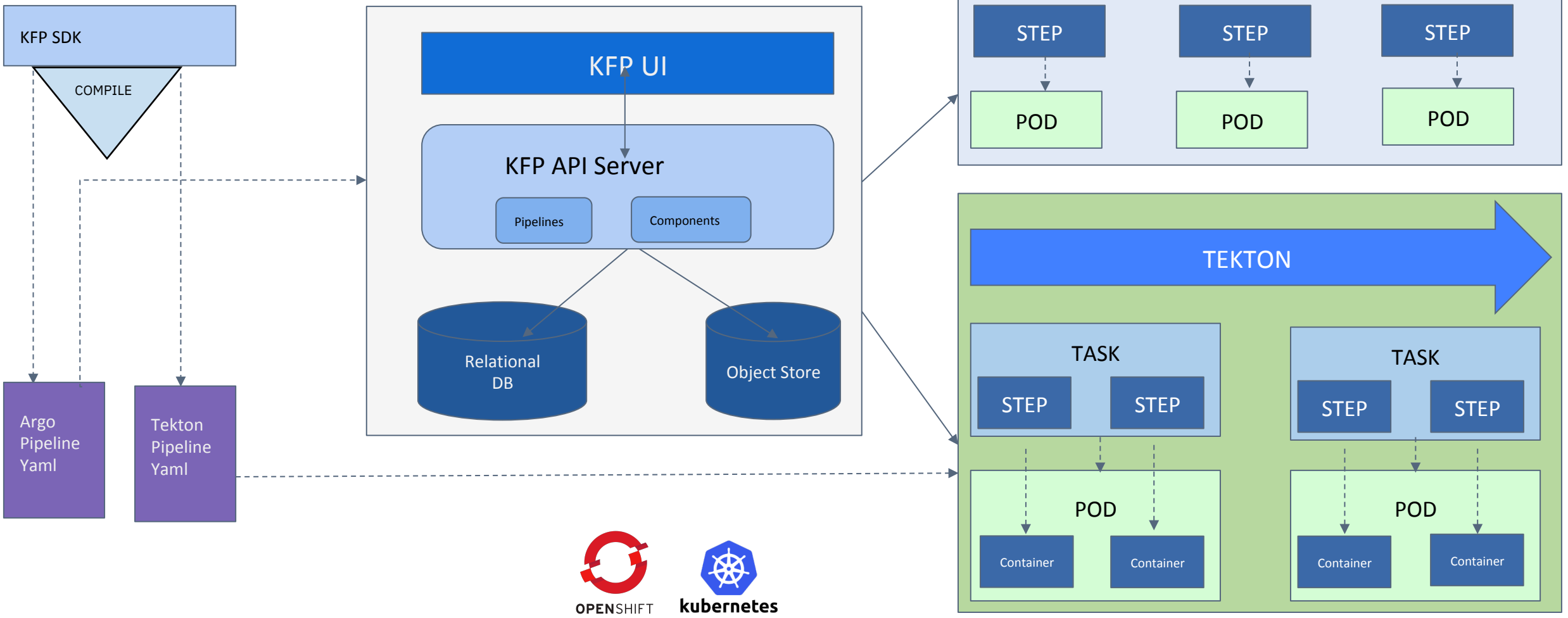
Tekton

- The Tekton Pipelines project provides Kubernetes-style resources for declaring CI/CD-style pipelines.
- Tekton introduces several new CRDs including Task, Pipeline, TaskRun, and PipelineRun.
- A PipelineRun represents a single running instance of a Pipeline and is responsible for creating a Pod for each of its Tasks and as many containers within each Pod as it has Steps.



- A **PipelineResource** defines an object that is an input (such as a git repository) or an output (such as a docker image) of the pipeline.
- A **PipelineRun** defines an execution of a pipeline. It references the Pipeline to run and the PipelineResources to use as inputs and outputs.
- A **Pipeline** defines the set of Tasks that compose a pipeline.
- A **Task** defines a set of build Steps such as compiling code, running tests, and building and deploying images.

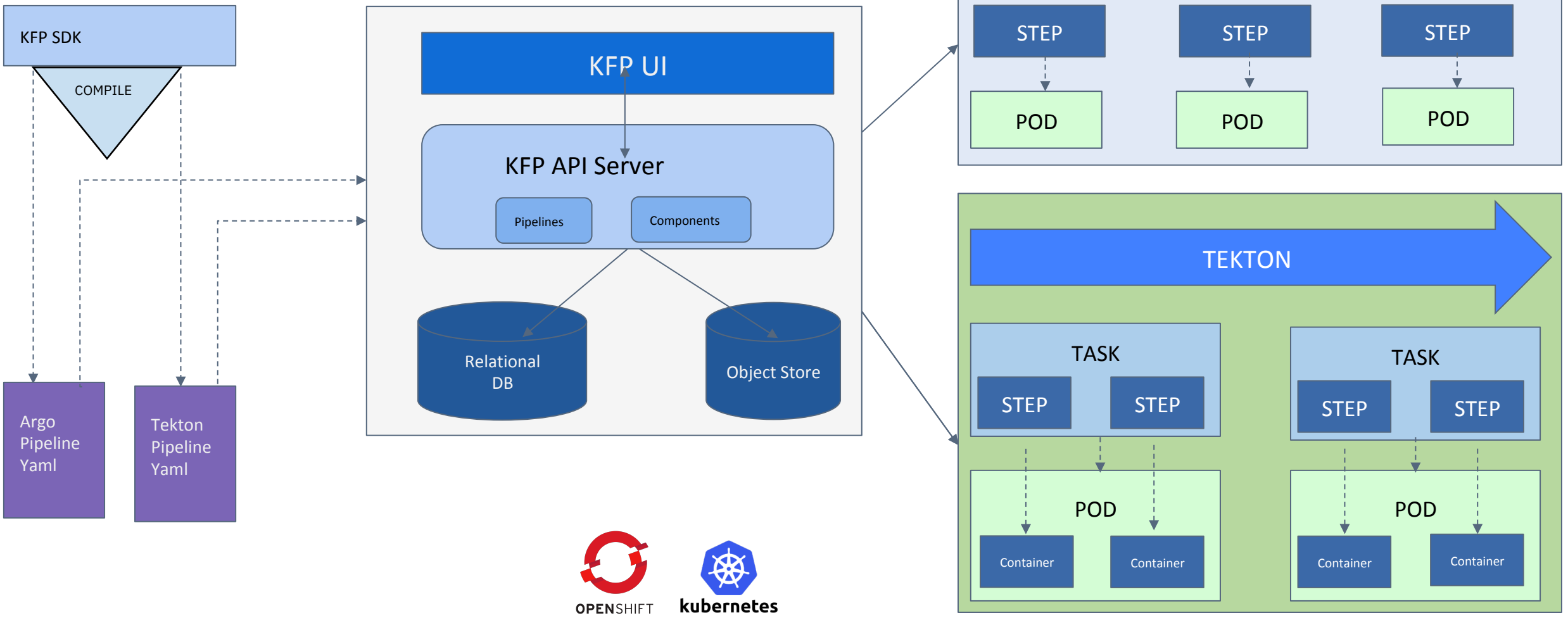
KFP – Tekton Phase One



Pluggable Components

Spark | Watson Studio | WML | Open Scale | Kubeflow Training | Seldon | AIF360 | ART | KATIB | KFSERVING | ... ...

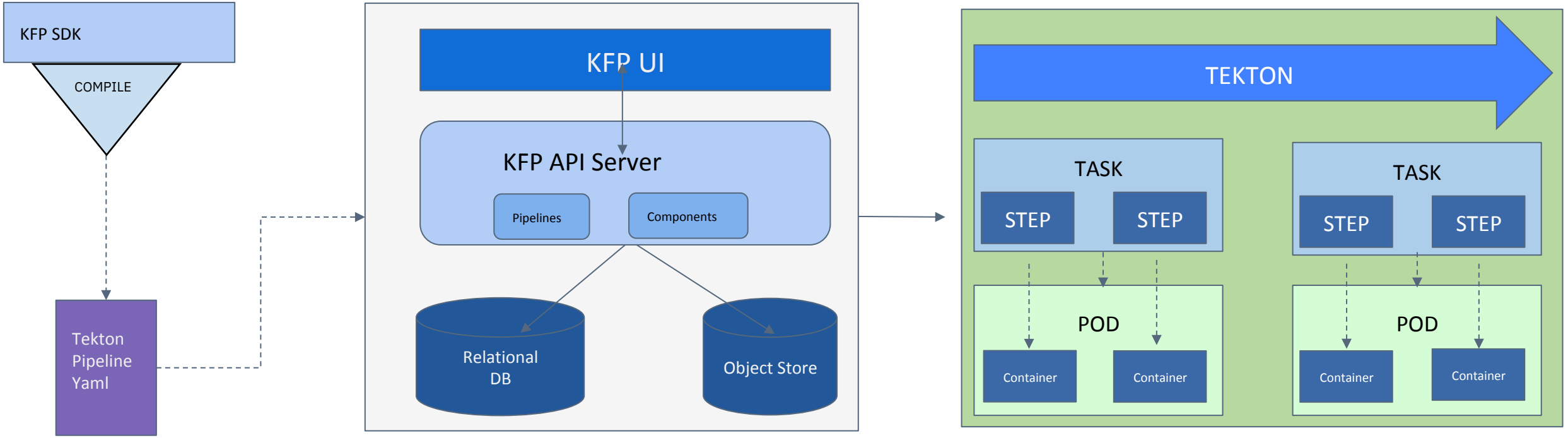
KFP – Tekton Phase Two



Pluggable Components

Spark | Watson Studio | WML | Open Scale | Kubeflow Training | Seldon | AIF360 | ART | KATIB | KFSERVING | ...

KFP – Tekton: Current Implementation

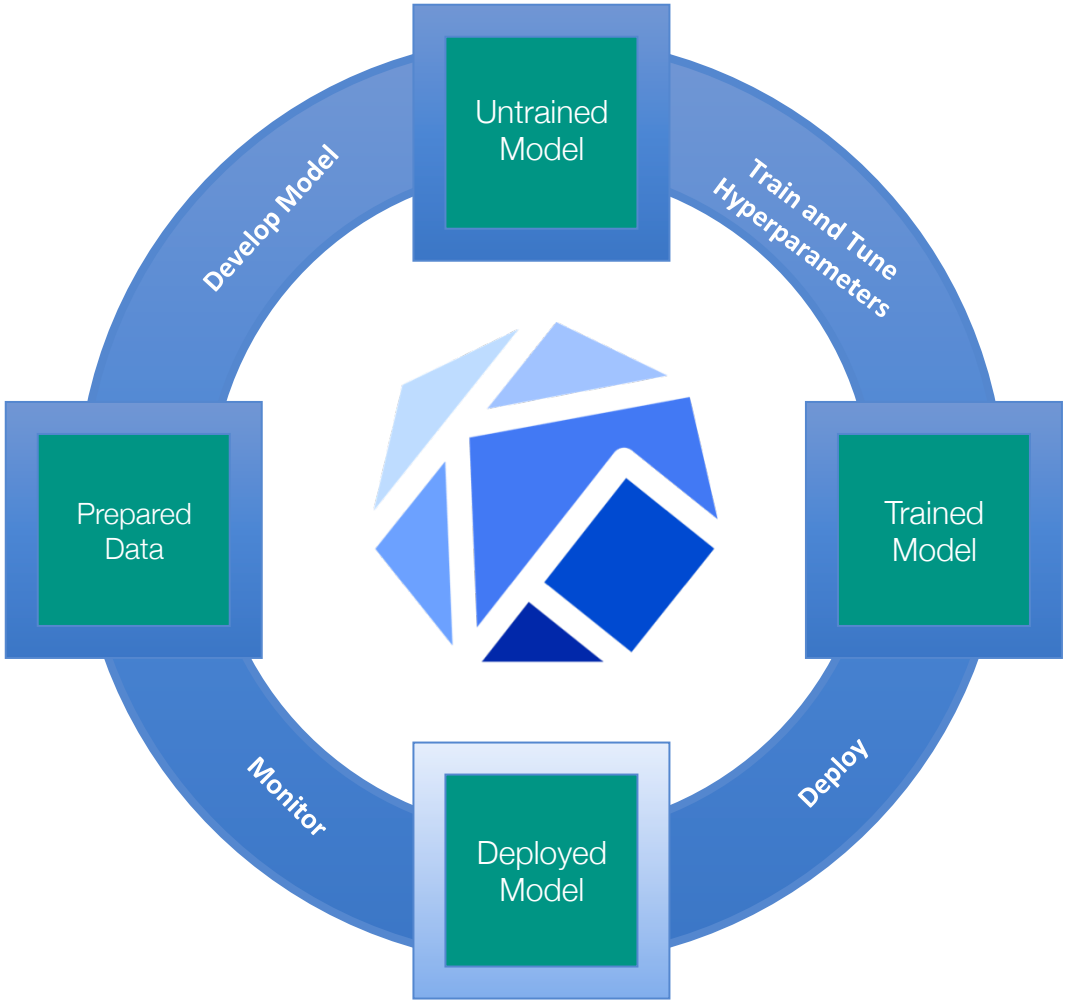


OPENSHIFT kubernetes

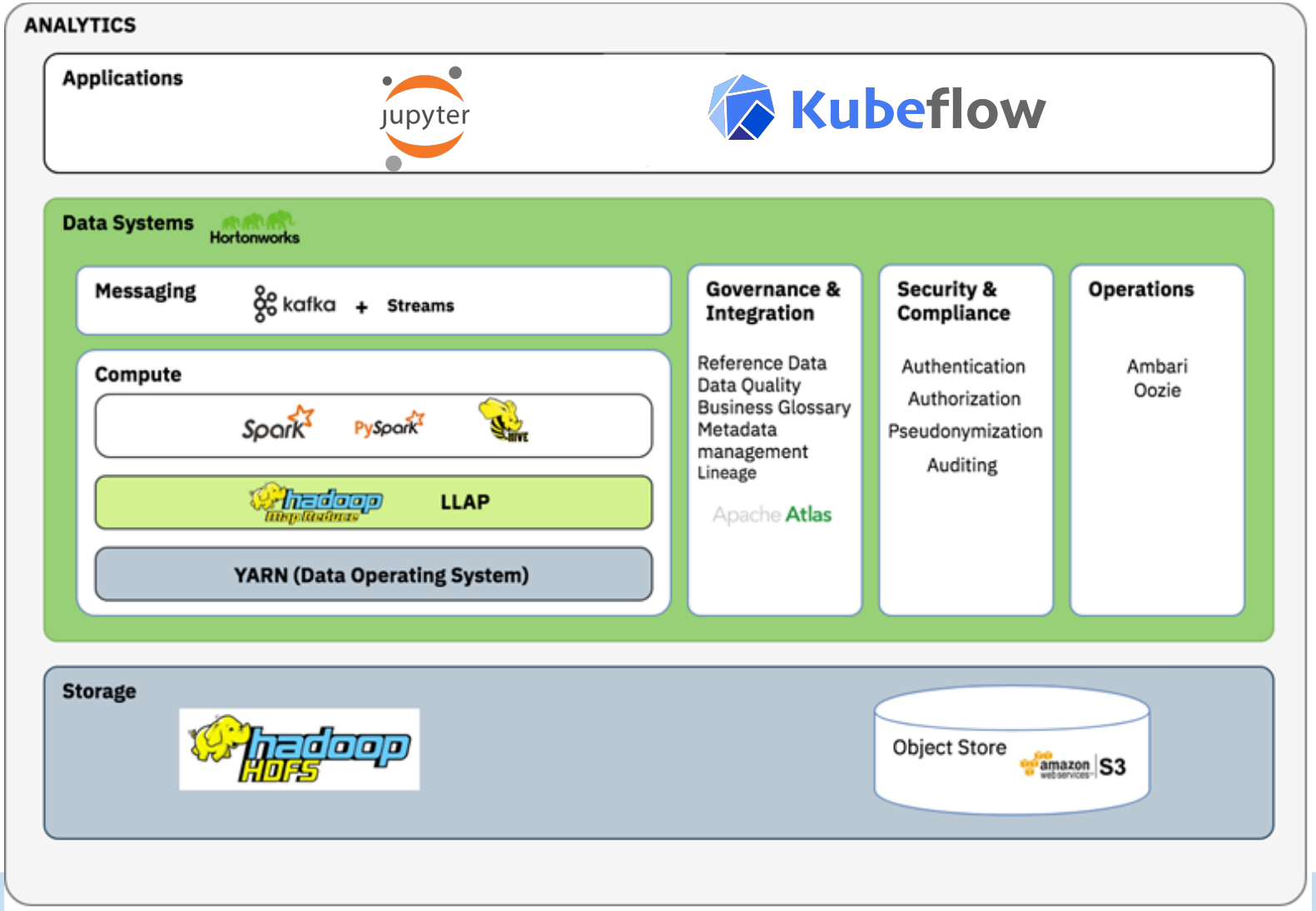
Pluggable Components

Spark | Watson Studio | WML | Open Scale | Kubeflow Training | Seldon | AIF360 | ART | KATIB | KFSERVING | ...

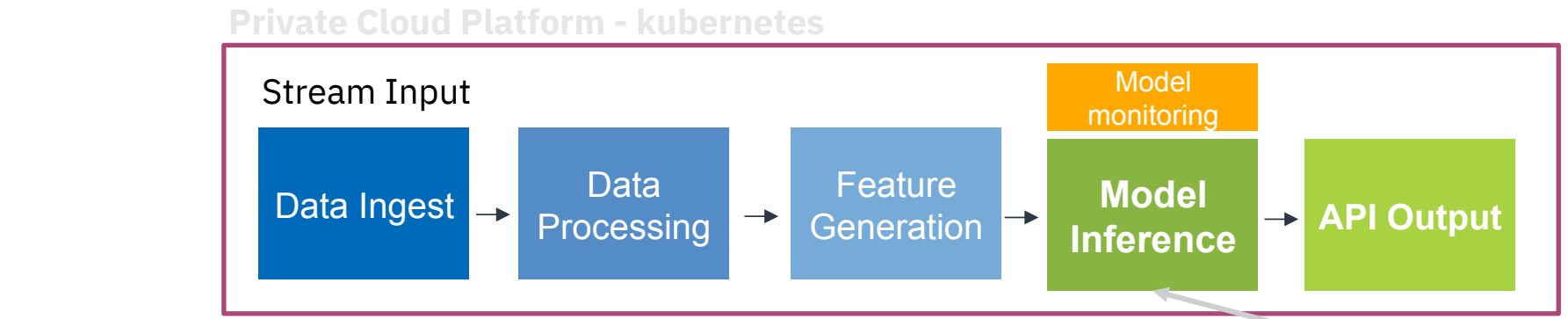
Kubeflow Adoption: External and Internal



Yara – Working with IBM to build a Data Science Platform for Digital Farming ML u cases based on Kubeflow

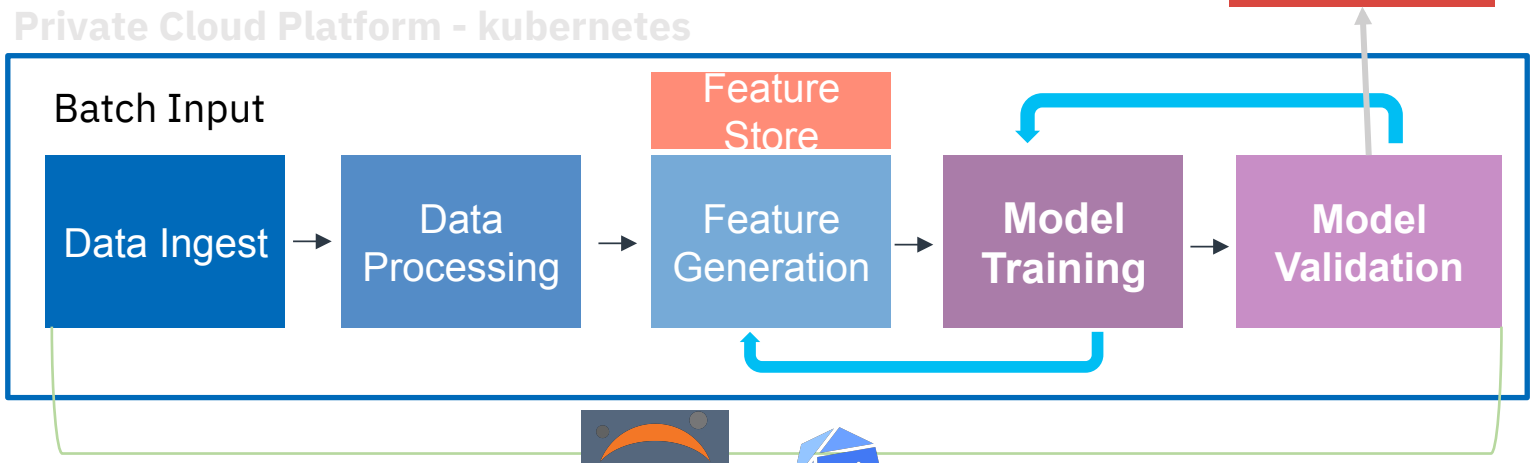


Telstra: Collaborating with IBM to build an Open Source based OneAnalytics Platform leveraging Kubeflow



Kubeflow

Model Deployment



jupyter

Kubeflow Pipelines

Model Training*

Katib

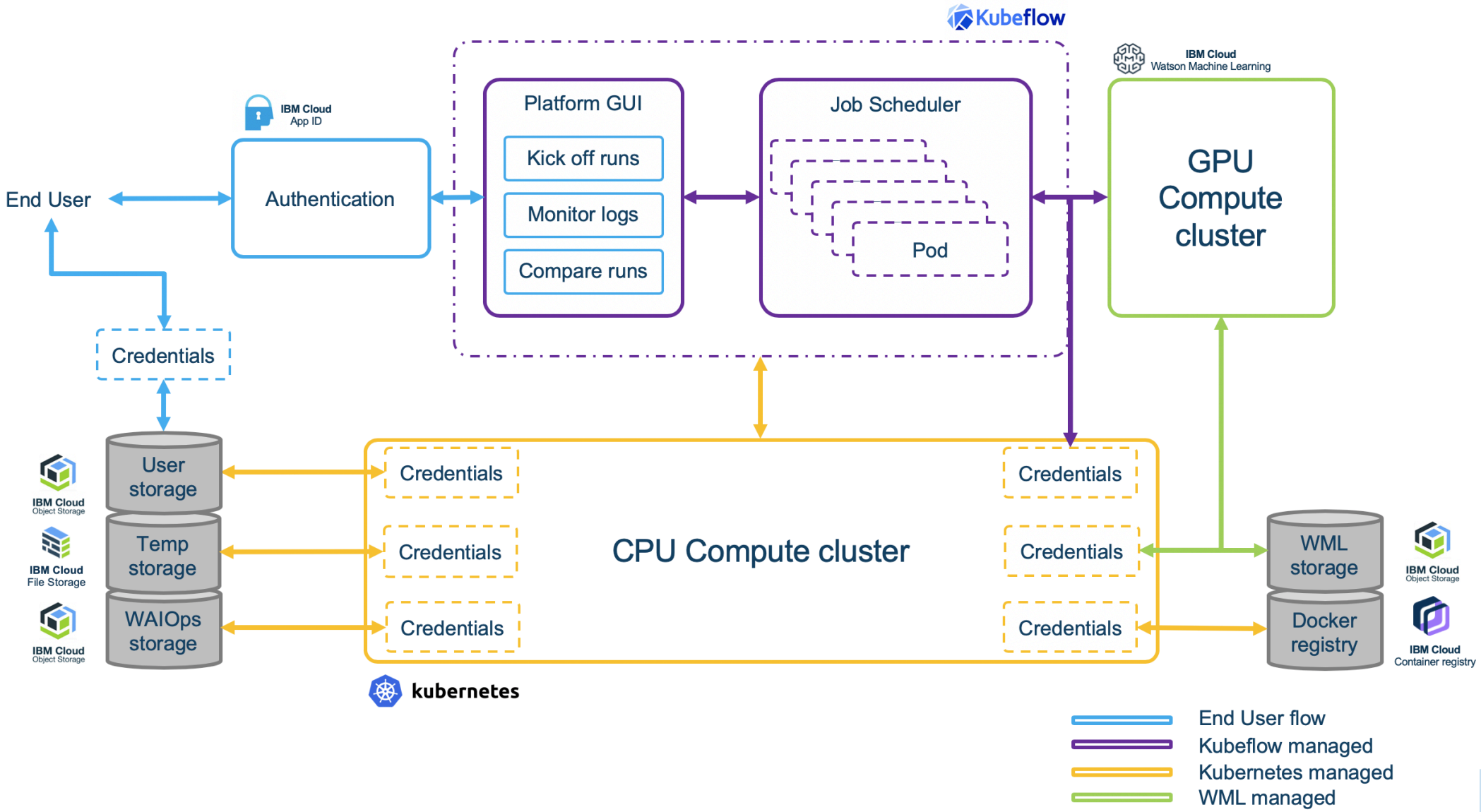
Model Serving*

SELDON

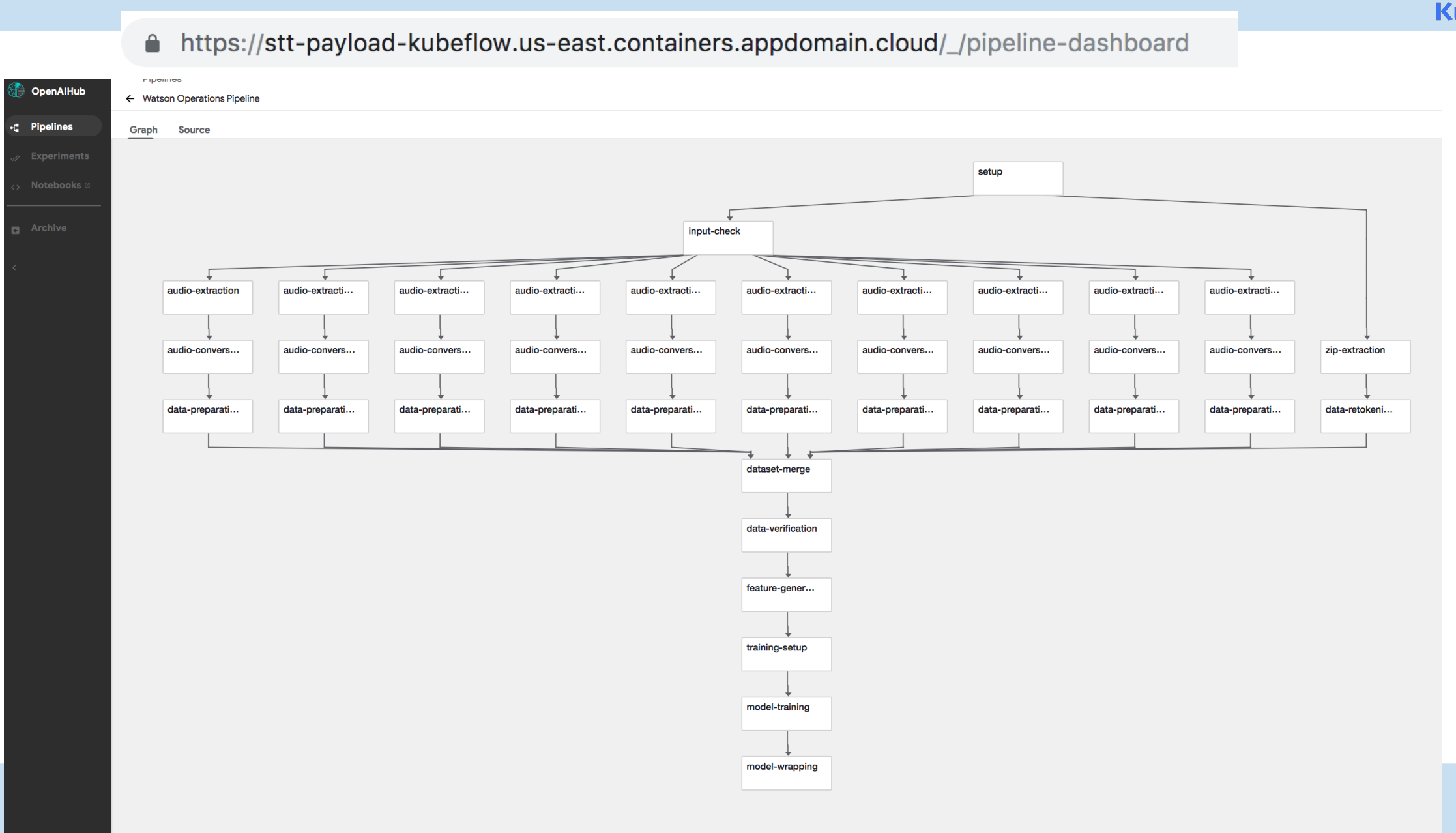
Kubeflow KFserving

*IBM PowerAI Docker image (https://hub.docker.com/r/ibmcom/powerai provide python libraries for both x86 and POWER systems

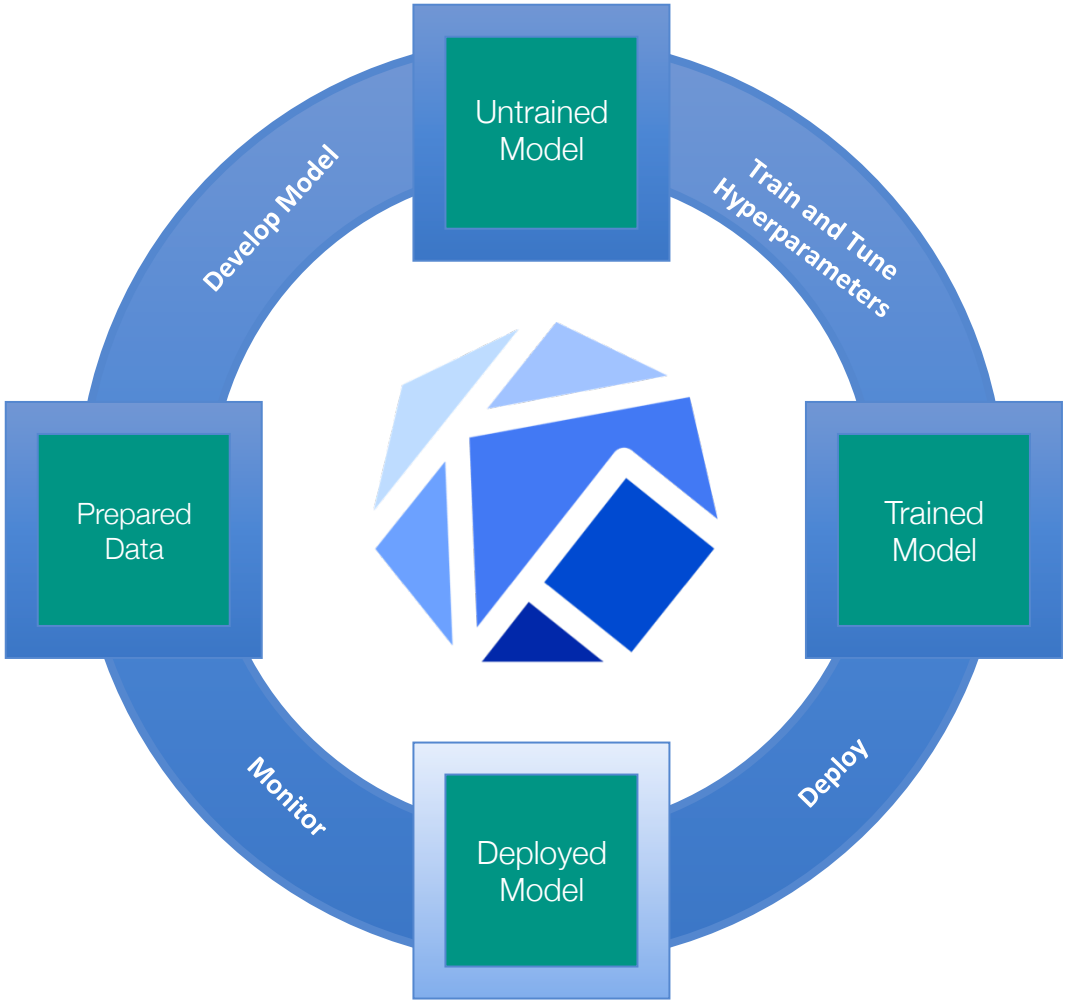
Watson AI Operations: Kubeflow Pipelines in production

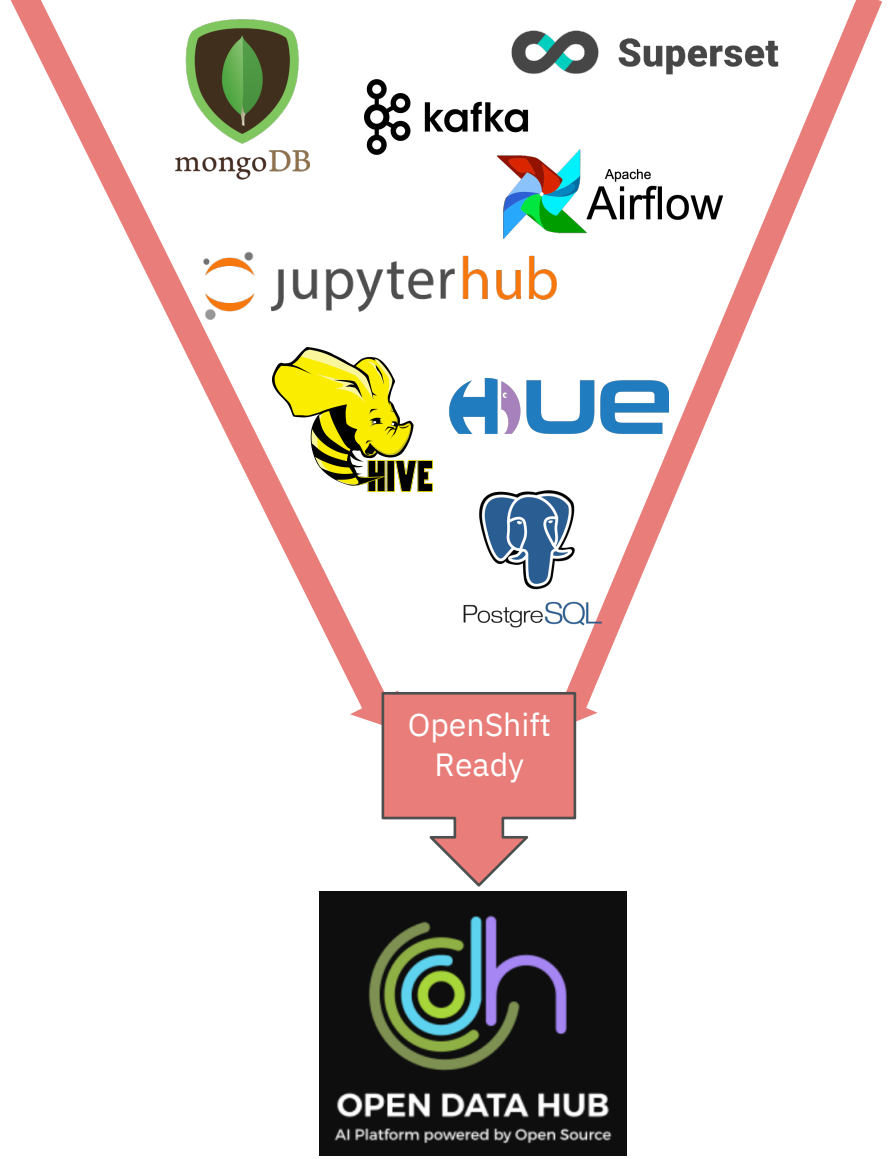


Watson SpeechToText training Kubeflow pipeline



OpenDataHub

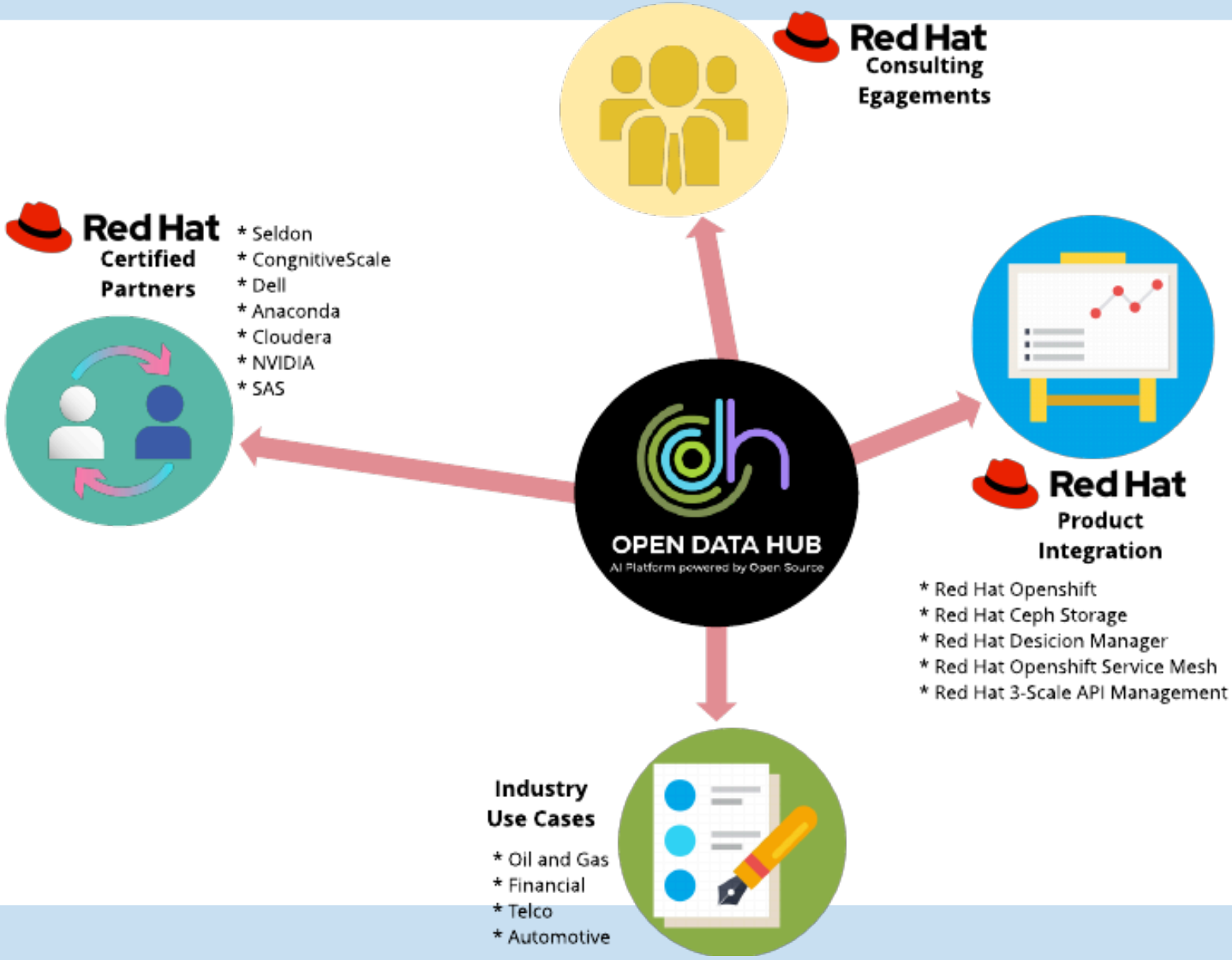




Data Platform

Operator Hub - operatorhub.io

OPEN DATA HUB - Ecosystem



OPEN DATA HUB
REFERENCE ARCHITECTURE

Kubeflow

AI and ML
- Interactive Notebooks
- Model Lifecycle
- ML Applications
- Business Applications

Data Analysis
- Big Data Processing
- Streaming
- Data Exploration

Metadata Management
- Metastore

Storage
- Data Lake
- In Memory
- Relational Databases

Data in Motion
- Streaming Data
- Object Storage Data
- Log Data

Security and Governance

Monitoring and Orchestration

Red Hat
OpenShift Container Platform

OPEN DATA HUB
REFERENCE IMPLEMENTATION

Kubeflow

AI and ML

- **Interactive Notebooks**: JupyterHub, Hue
- **Model Lifecycle**: Kubeflow, Seldon, MLFlow
- **ML Applications**: OpenDataHub AI Library
- **Business Applications**: Superset

Data Analysis

- **Big Data Processing**: Spark, SparkSQL, Thrift
- **Streaming**: Kafka Streams, Elastic Search
- **Data Exploration**: Hue, Kibana

Metadata Management

- **Metastore**: Hive

Storage

- **Data Lake**: RedHat Ceph Storage
- **In Memory**: RedHat Data Grid (Infinispan)
- **Relational Databases**: PostgreSQL, MySQL

Data in Motion

- **Streaming Data**: RedHat AMQ Streams, Kafka Connect
- **Object Storage Data**: RedHat Ceph S3 API
- **Log Data**: FluentD, Logstash

Red Hat
OpenShift Container Platform

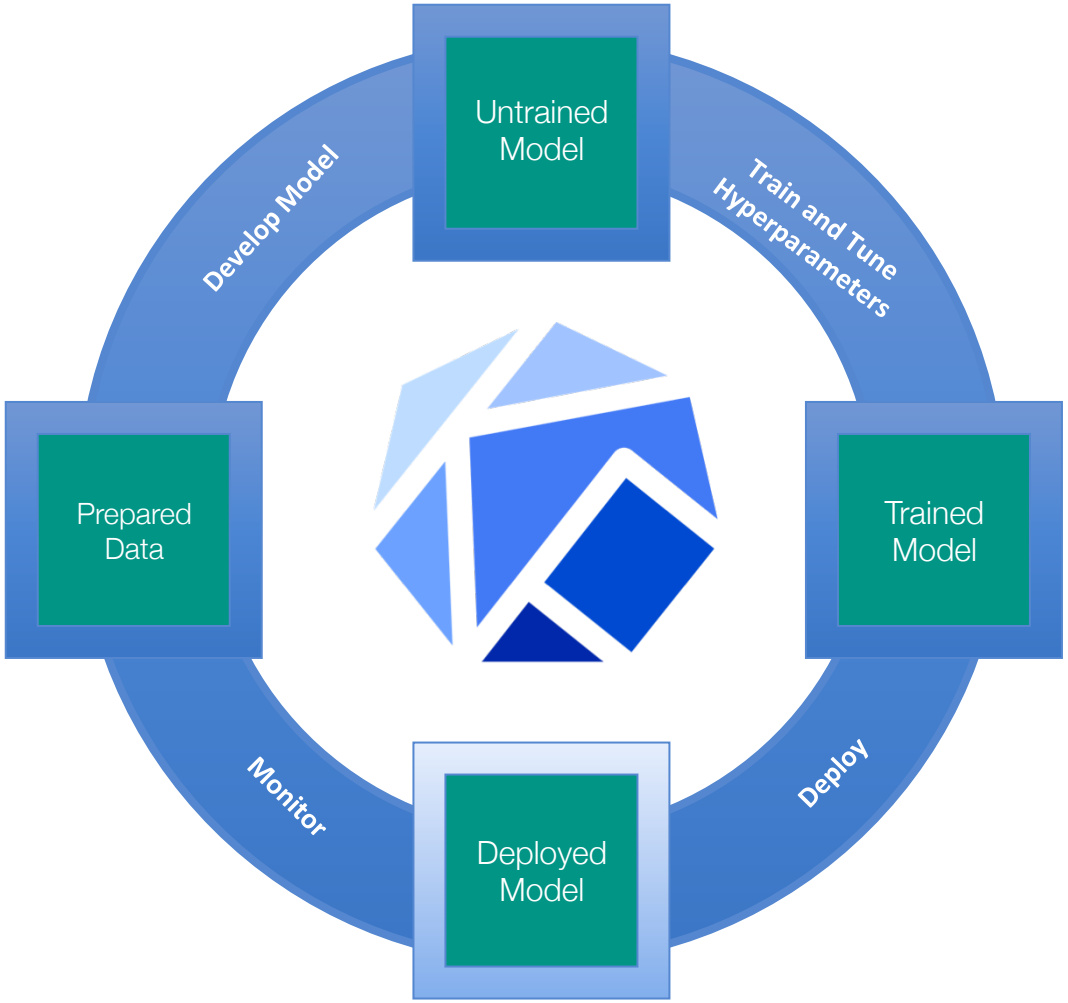
Security and Governance

- OpenShift Oauth
- OpenShift Single SignOn (Keycloak)
- RedHat Ceph Object Gateway
- RedHat 3scale

Monitoring and Orchestration

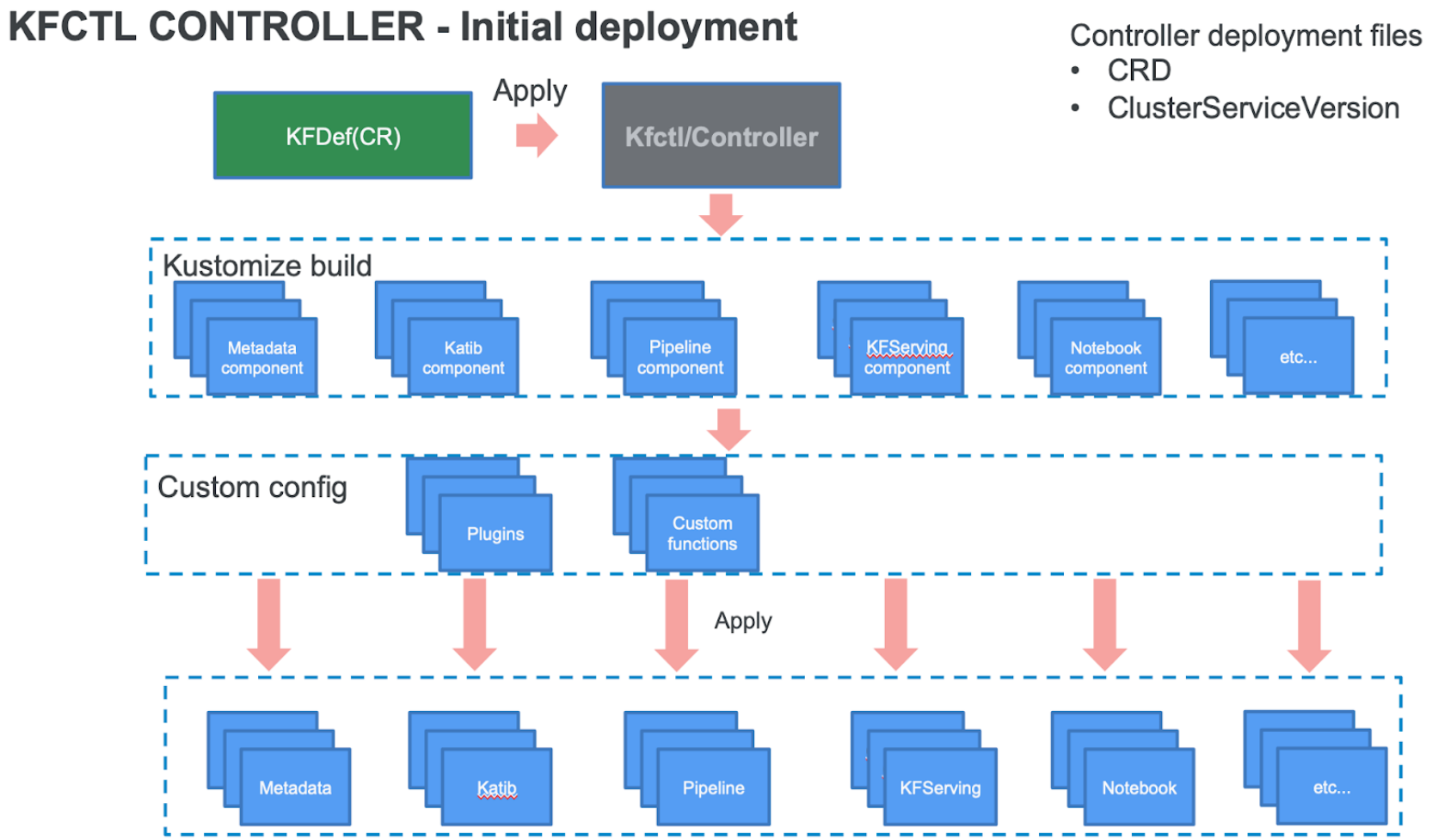
- Prometheus
- Grafana
- Kubeflow Pipelines
- Jenkins CI/CD

OpenDataHub and Kubeflow: Relationship



Kubeflow Operator – Contributed by IBM to Kubeflow community

- Deploy, manage and monitor Kubeflow
- On various environments
  - ❑ IBM Cloud
  - ❑ GCP
  - ❑ AWS
  - ❑ Azure
  - ❑ OpenShift
  - ❑ Other K8S



ODH and Kubeflow: Kubeflow an Upstream for OpenDataHub

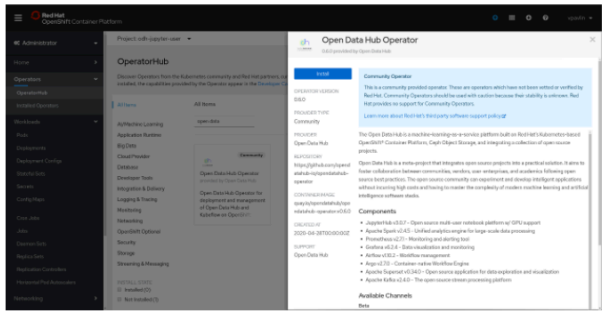
- An version of the Operator based on Kubeflow Architecture released- https://developers.redhat.com/blog/2020/05/07/open-data-hub-0-6-brings-component-updates-and-kubeflow-architecture/?sc_cid=7013a000002DTqEAAW
- Most of the components converted https://github.com/opendatahub-io/odh-manifests
- Still a separate deployment – needs to do both ODH and Kubeflow in one go.

Future

- KF 1.0 on OpenShift
- Disconnected deployment
- Open Data Hub CI/CD
- Kubeflow on OpenShift CI
- UBI based ODH & KF
- Multitenancy model
- Mixing KF & ODH

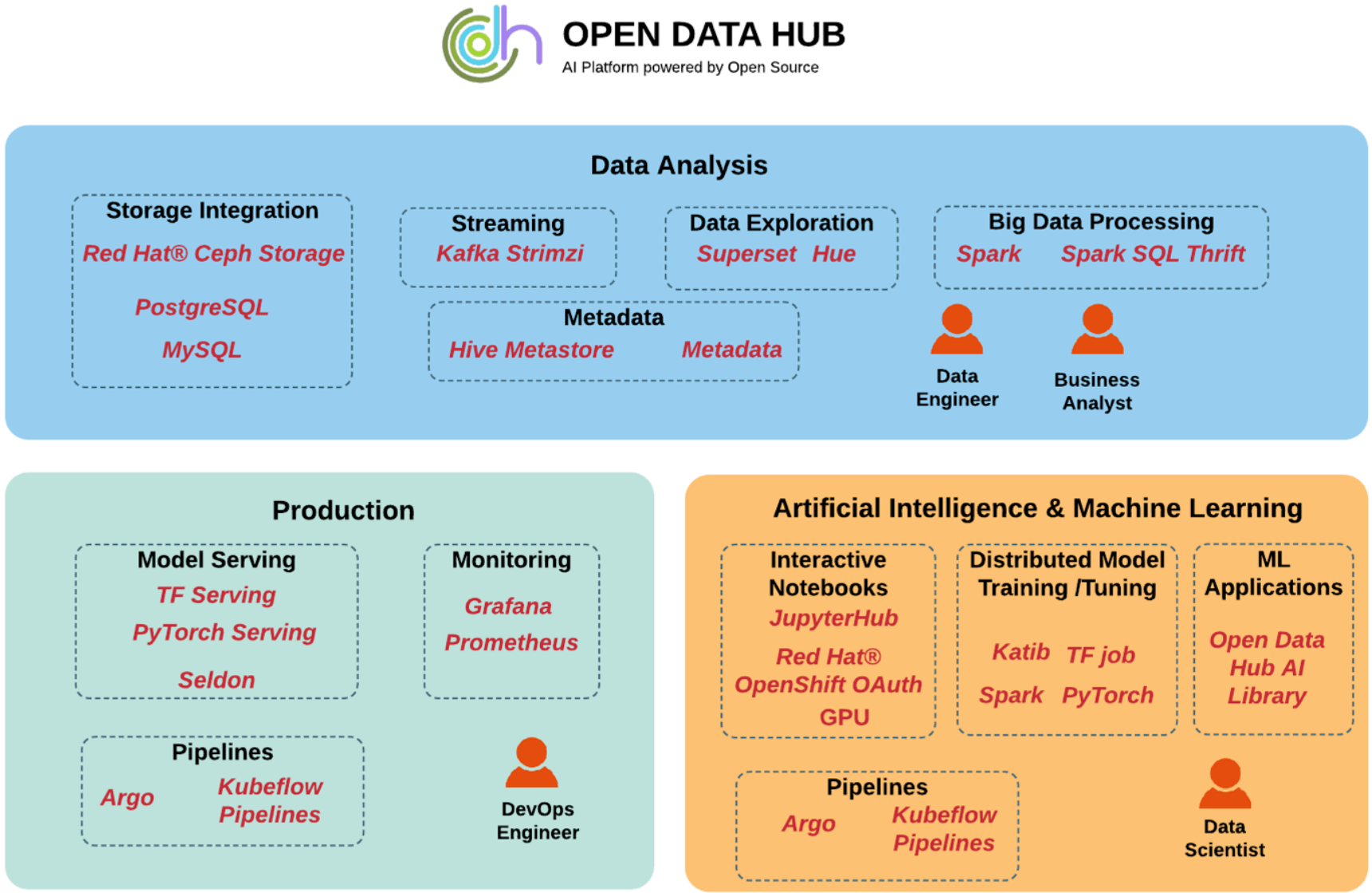
Open Data Hub 0.6 brings component updates and Kubeflow architecture

By Václav Pavlín May 7, 2020

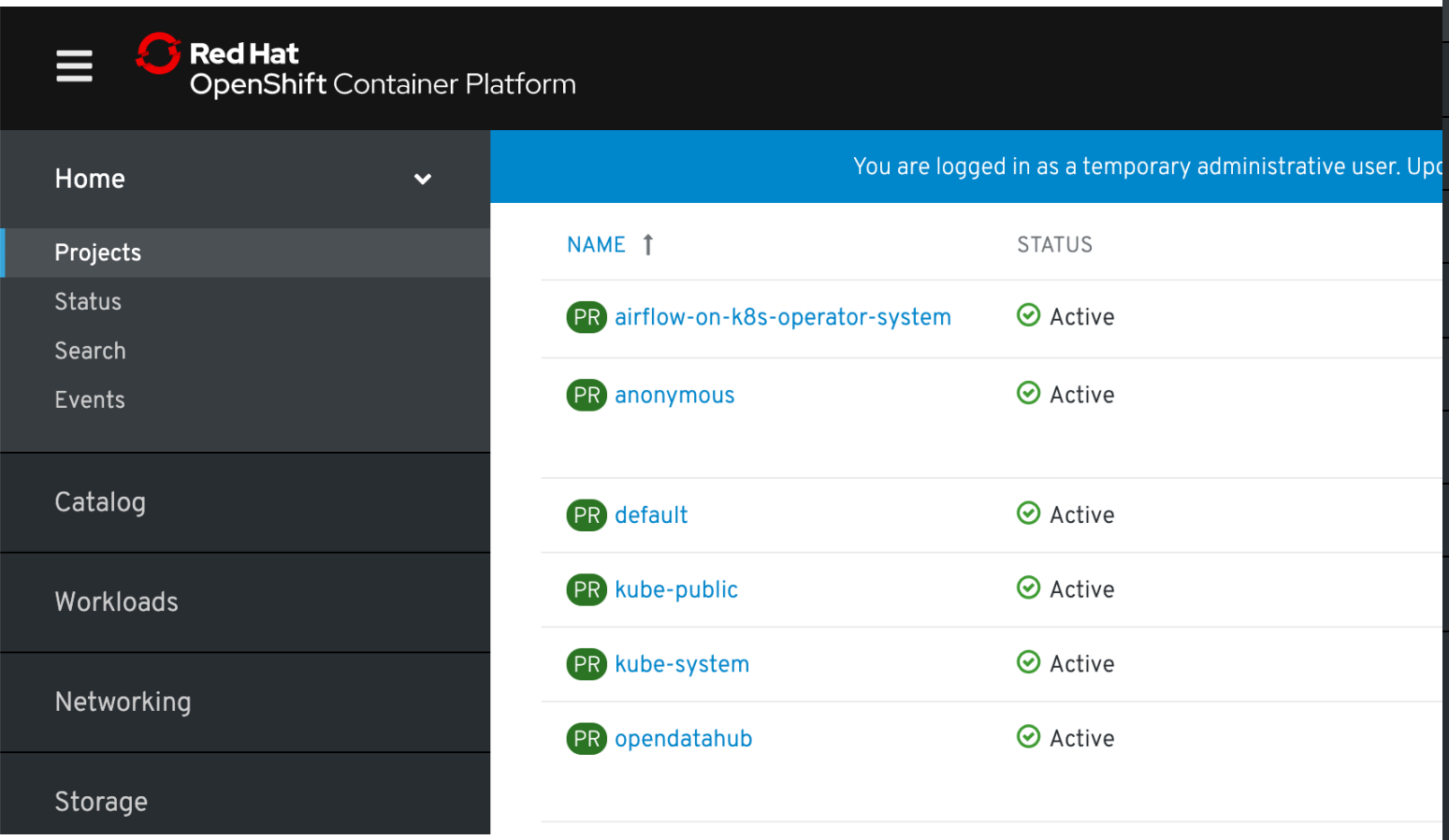


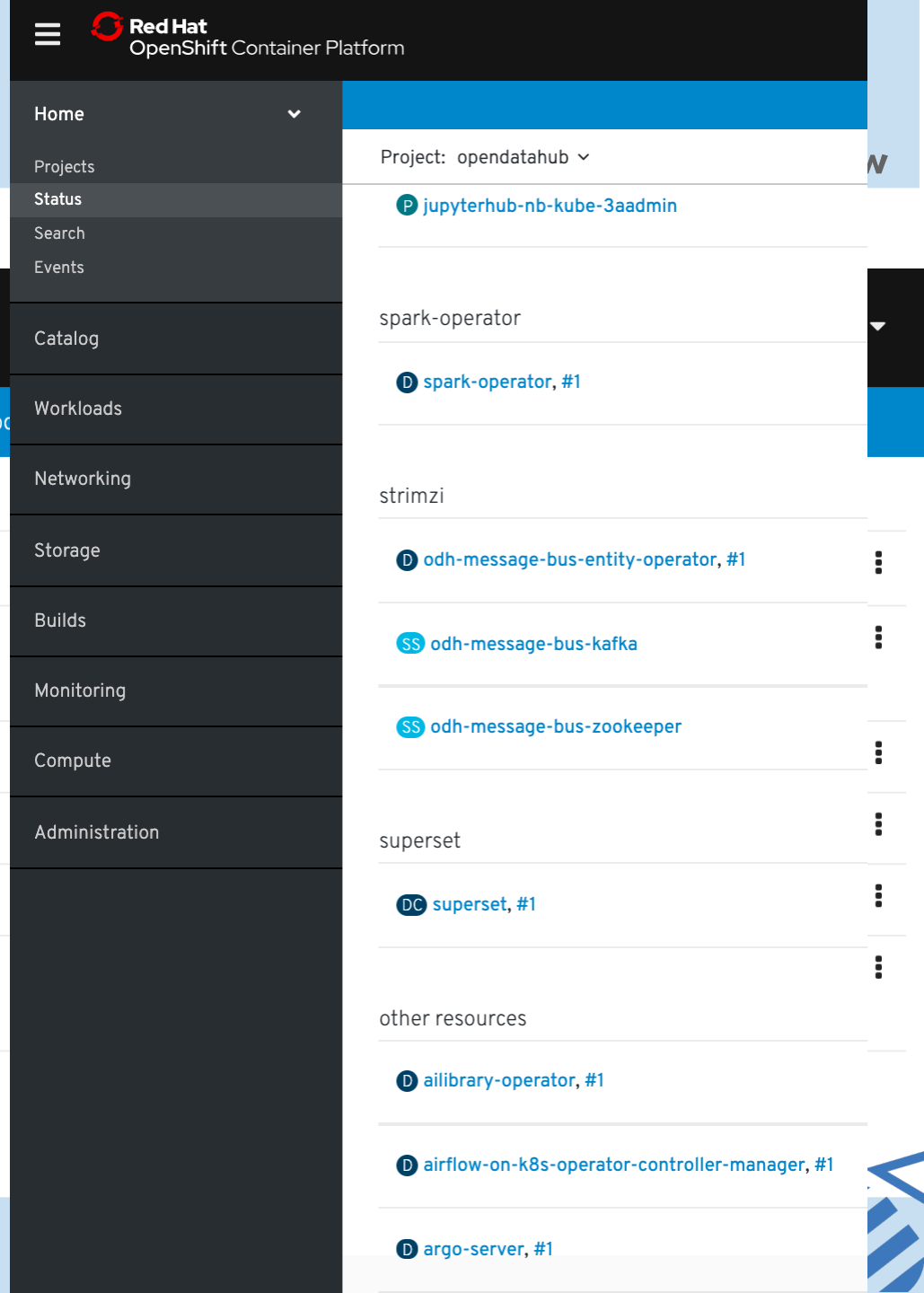
Open Data Hub (ODH) is a blueprint for building an AI-as-a-service platform on Red Hat's Kubernetes-based OpenShift 4.x. Version 0.6 of Open Data Hub comes with significant changes to the overall architecture as well as component updates and additions. In this article, we explore these changes.

OPEN DATA HUB 0.6.x



Open Data Hub in OpenShift





IBM Apache Superset

Superset | Security | Manage | Sources | Charts | Dashboards | SQL Lab

Growth Analysis | Scratchpad

Database: main

Schema: superset

Add a table (43)

slices

created_on	DATETIME
changed_on	DATETIME
id	INTEGER
slice_name	VARCHAR
datasource_type	VARCHAR
datasource_name	VARCHAR
viz_type	VARCHAR
params	TEXT
created_by_fk	INTEGER
changed_by_fk	INTEGER
description	TEXT
cache_timeout	INTEGER
perm	VARCHAR
datasource_id	INTEGER

dashboards

created_on	DATETIME
changed_on	DATETIME
id	INTEGER
dashboard_title	VARCHAR
position_json	TEXT
created_by_fk	INTEGER
changed_by_fk	INTEGER
css	TEXT

```
1 SELECT b.dashboard_id, a.dashboard_title, b.slice_id, c
2 FROM dashboards a
3 JOIN dashboard_slices b ON a.id = b.das
4 JOIN slices c on c.id = b.slice_id
```

dashboards
dashboard_title
DATABASE
datasource_type
datasource_name
datasource_id

Run Query | Save Query | Share Query

Results | Query History | Preview for slices | Preview for d

Visualize | .CSV

dashboard_id	dashboard_title	slice_id	slice_name
2	Births	882	Girls
2	Births	883	Boys
2	Births	884	Participants
2	Births	885	Genders
2	Births	886	Genders by State
2	Births	887	Trends
2	Births	888	Average and Sum Tre
2	Births	889	Title
2	Births	890	Name Cloud

Superset | Security | Manage | Sources | Charts | Dashboards | SQL Lab

World's Bank Data

Edit Dashboard | Actions

Region Filter

Since | Until

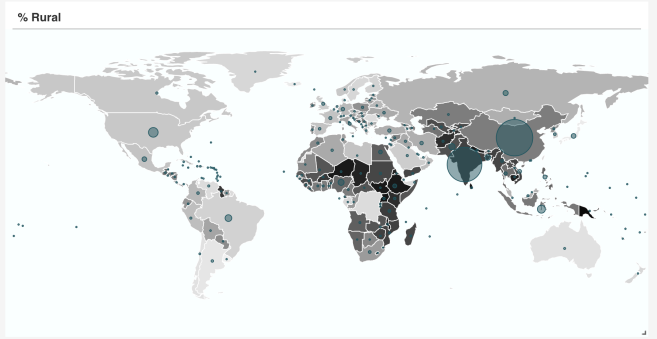
region: Select [region]

country_name: Select [country_name]

World's Population

7.24G

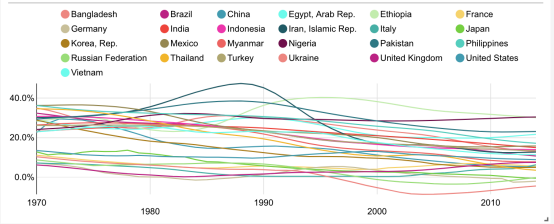
% Rural



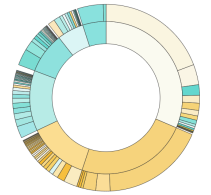
Most Populated Countries

country_name	sum__SP_POP_TOTL
China	1.36G
India	1.30G
United States	319M
Indonesia	254M
Brazil	206M
Pakistan	185M
Nigeria	177M
Bangladesh	159M
Russian Federation	144M
Japan	127M
Mexico	125M
Philippines	99.1M
Ethiopia	97.0M
Vietnam	90.7M
Egypt, Arab Rep.	89.6M
Germany	80.9M
Iran, Islamic Rep.	78.1M
Turkey	75.9M
Congo, Dem. Rep.	74.9M
Thailand	67.7M
France	66.2M
United Kingdom	64.5M
Italy	61.3M
South Africa	54.0M
Myanmar	53.4M
Tanzania	51.8M
Korea, Rep.	50.4M
Colombia	47.8M
Spain	46.4M
Ukraine	45.4M
Kenya	44.9M
Argentina	43.0M
Sudan	39.4M
Algeria	38.9M
Poland	38.0M
Uganda	37.8M

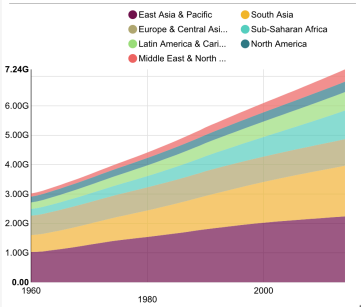
Growth Rate



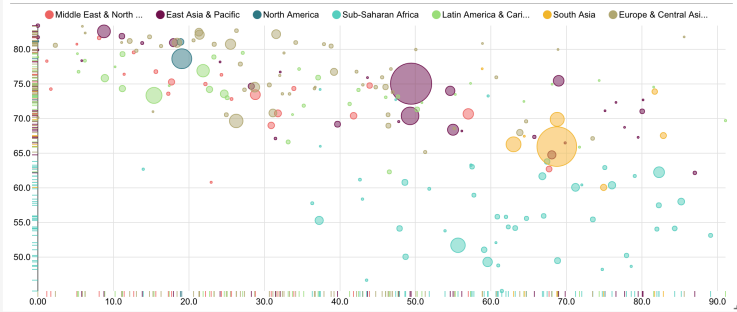
Rural Breakdown



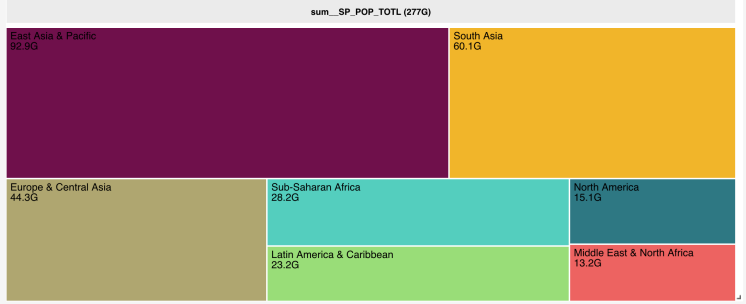
World's Pop Growth



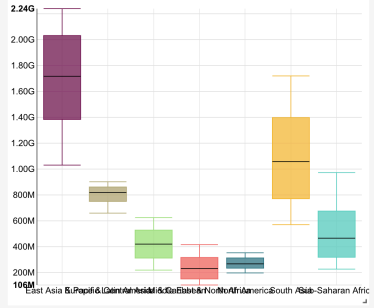
Life Expectancy VS Rural %



Treemap

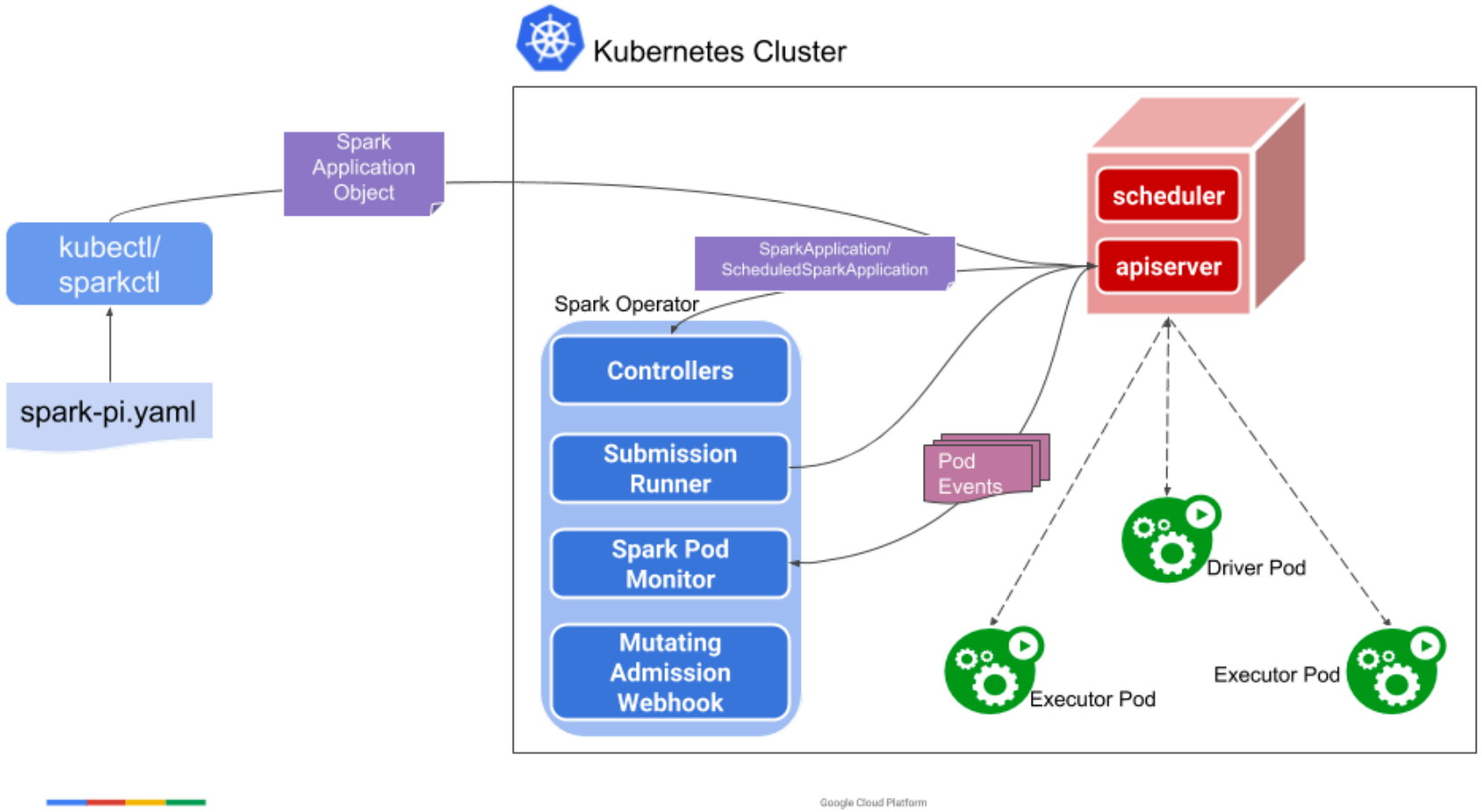


Box plot



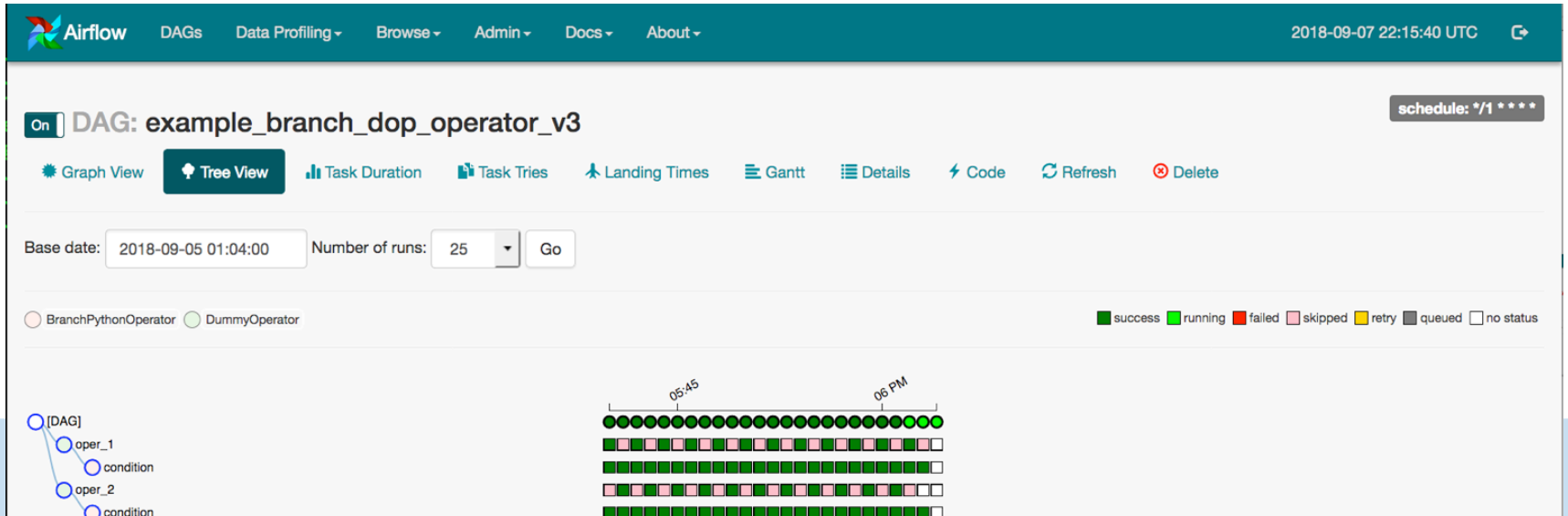
Spark with Open Data Hub

- Open Data Hub will also deploy the Spark Operator to manage Spark as an application.
- Two versions of Spark – Spark in dedicated mode and Spark on K8s
- Currently moving towards Spark on K8s Operator from Google for serverless Spark. IBM Hummingbird team investigating this



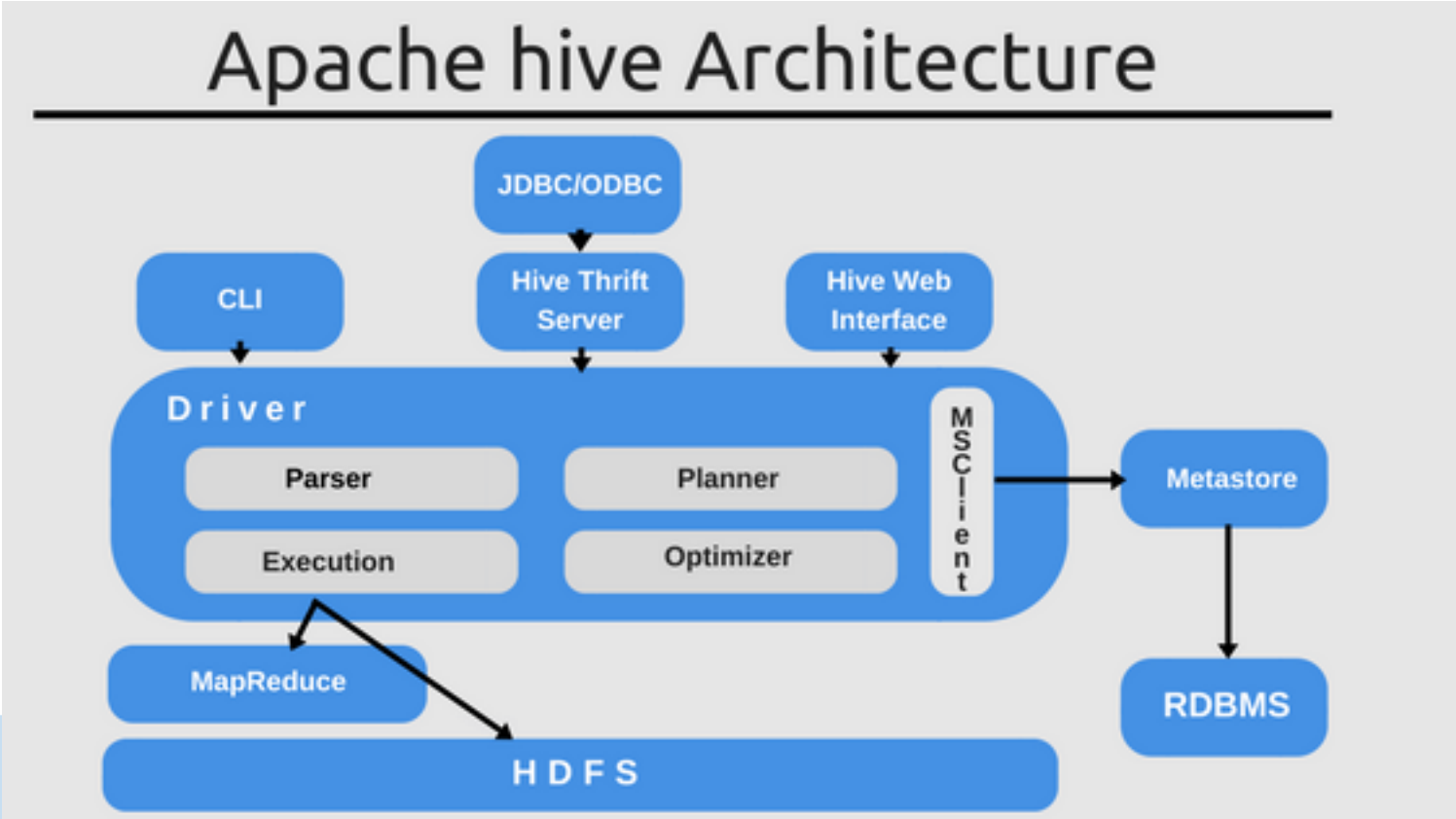
Airflow integration with Open Data Hub

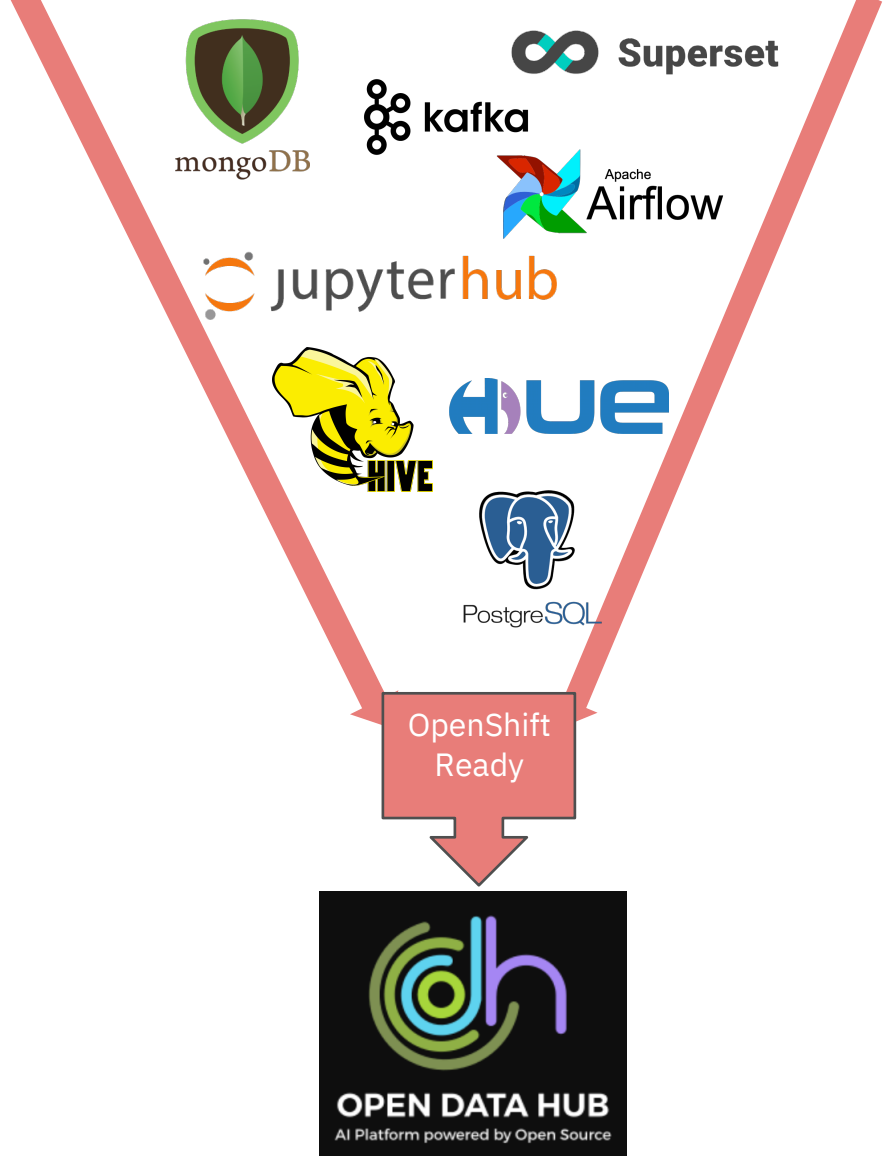
- Open Data Hub will also deploy the Airflow Operator to manage Airflow as an application.
- Using the Airflow Operator originally developed in the GoogleCloudPlatform repository and later donated to Apache.
- The Operator creates a controller-manager pod which will be created as a part of the Open Data Hub deployment.
- Users can then install the Airflow components they need from the available options (eg: CeleryExecutor or KubernetesExecutor, Postgres deployment or MySQL deployment etc.)



Apache Hive with OpenDataHub

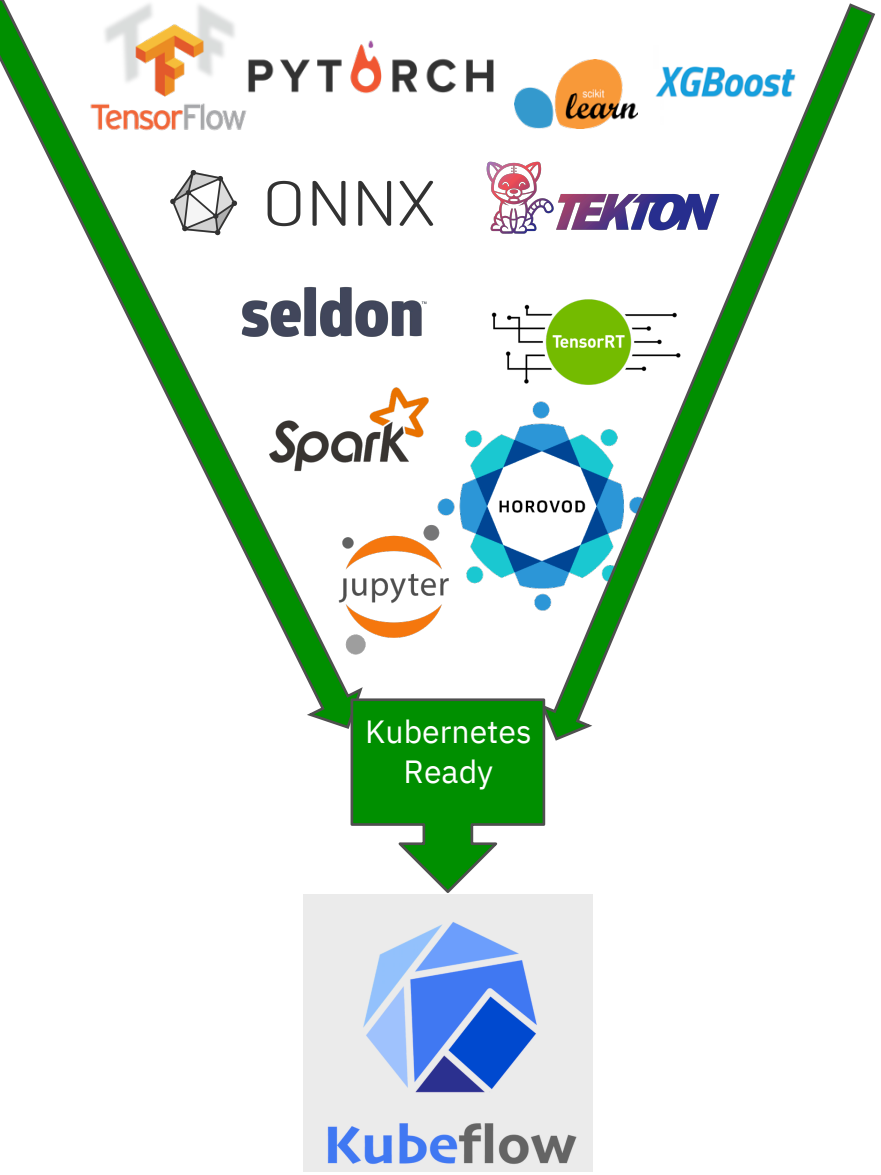
- Hive was one of the first abstraction engines to be built on top of MapReduce.
- Started at Facebook to enable data analysts to analyse data in Hadoop by using familiar SQL syntax without having to learn how to write MapReduce.
- Hive an essential tool in the Hadoop ecosystem that provides an SQL dialect for querying data stored in HDFS, other file systems that integrate with Hadoop such as MapR-FS and Amazon's S3 and databases like HBase(the Hadoop database) and Cassandra.
- Hive is a Hadoop based system for querying and analysing large volumes of structured data which is stored on HDFS.
- Hive is a query engine built to work on top of Hadoop that can compile queries into MapReduce jobs and run them on the cluster.



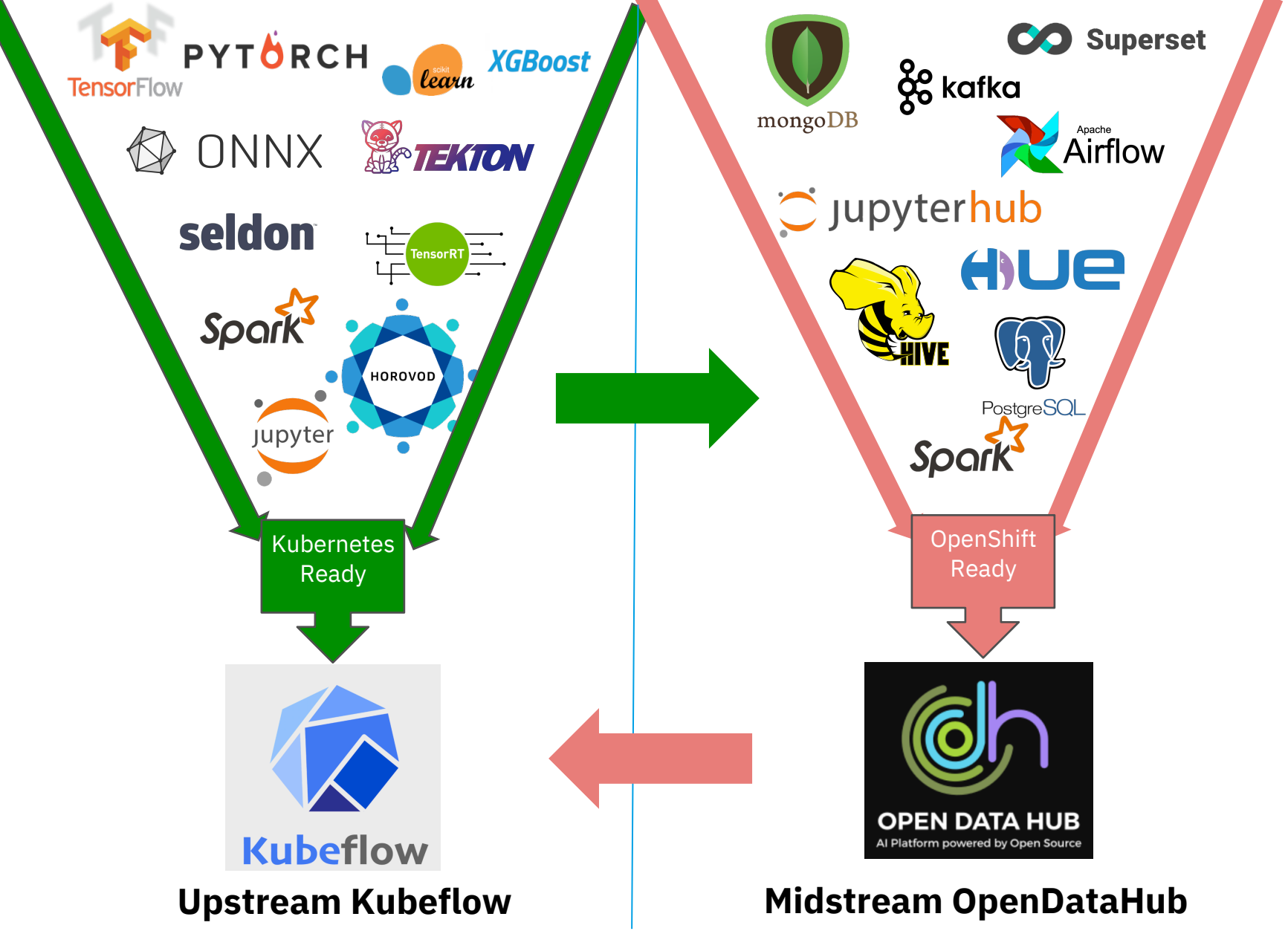


Data Platform

Operator Hub - operatorhub.io



ML and AI Platform



OpenDataHub

\+

Kubeflow

Open Source End To End

Data and AI Platform

Operator Hub - operatorhub.io

RedHat MarketPlace https://marketplace.redhat.com/en-us

Kubeflow Dojo

Date: **Wed July 15, 2020**

Time	Topic	Presenter
8:00am - 8:30 am	Data and AI Open Source at CODAIT	Animesh
8:30am - 9:30 am	Kubeflow Overview - End to end ML on Kubernetes	Animesh
9:30am - 9:45am	Break	
9:45am - 10:45am	Git and Github	Tom & Morgan
10:45am - 11:00am	Break	
11:00am - 11:30am	Kubeflow development environment	Weiqiang
11:30am - 12:00 pm	Control plane deep dive	Weiqiang
12:00pm - 1:00pm	Lunch break	
1:00pm - 2:00pm	Kubeflow deployment handson	
2:00pm - 3:00pm	Tryout Kubeflow Components	Tommy
3:00pm - 4:00pm	Q&A	

https://github.com/IBM/KubeflowDojo

https://github.com/kubeflow

https://github.com/opendatahub-io

Date: **Thu July 16, 2020**

Time	Topic	Presenter
8:00am - 8:30am	Overview of Kubeflow repos	Tommy
8:30 am - 9:30am	Kubeflow Pipelines deep dive	Animesh, Tommy, Christian
9:30am - 9:45am	Break	
9:45 am - 10:45am	Kubeflow Pipelines-Tekton hands on	Christian Kadner, Tommy Li
10:45am - 11 am	Break	
11:00am - 12 am	KFServing deep dive	Animesh, Tommy
12:00pm - 1:00pm	Lunch break	
1:00pm - 2:00pm	Distributed Training and HPO Deep Dive	Andrew, Kevin, Animesh
2:00pm - 2:15pm	Break	
2:15pm - 2:30pm	Kubeflow PR workflow	Weiqiang
2:30pm - 3:30pm	PR workflow handson	
3:30pm - 4:00pm	Wrap up and final Q&A	Animesh