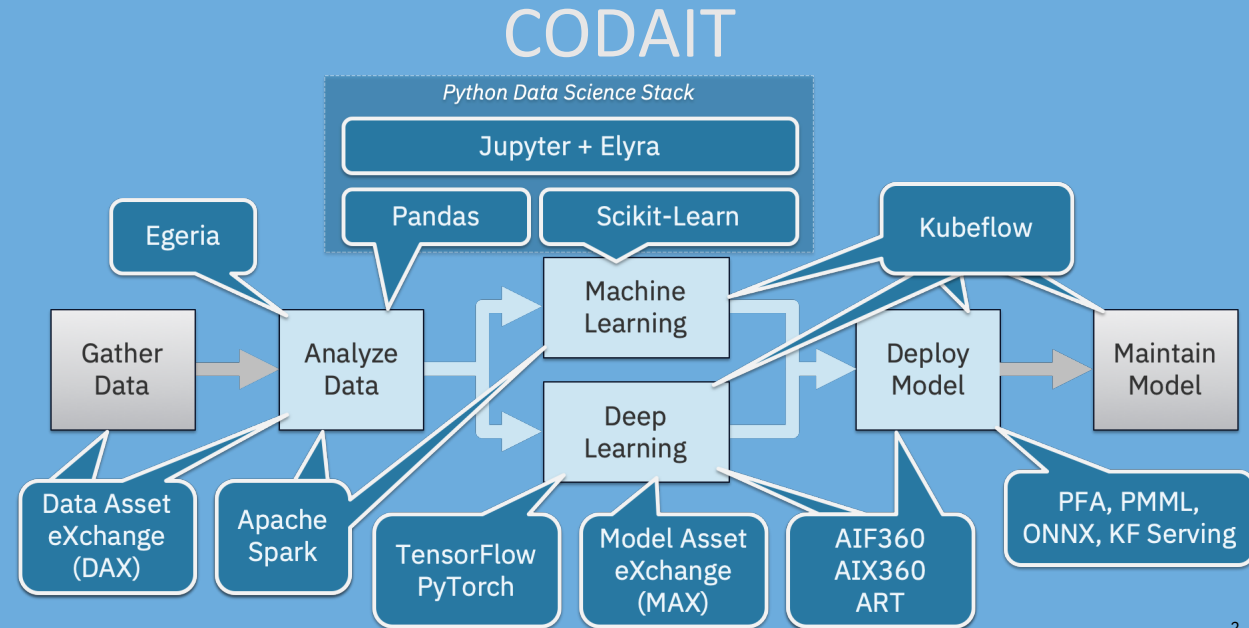


# Your Speaker Today:



## Animesh Singh

STSM and Chief Architect - Data and AI Open Source Platform

- CTO, IBM RedHat Data and AI Open Source Alignment
- IBM Kubeflow Engagement Lead, Kubeflow Committer
- Chair, Linux Foundation AI - Trusted AI
- Chair, CD Foundation MLOps Sig
- Ambassador, CNCF
- Member of IBM Academy of Technology (IBM AoT)

## Kubeflow

[github.com/kubeflow](https://github.com/kubeflow)

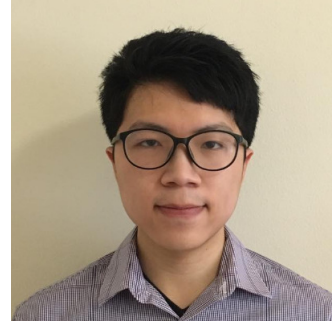
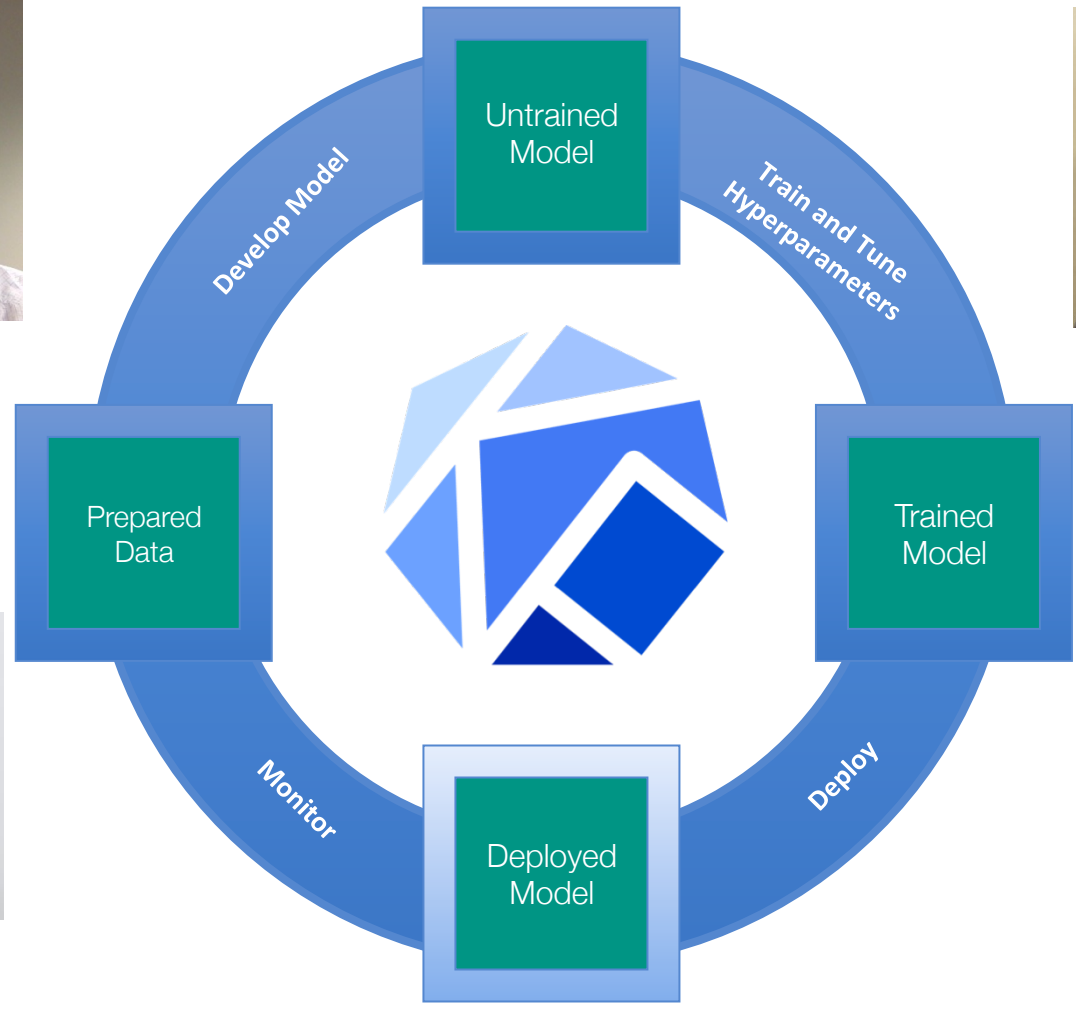




Christian Kadner



Weiqiang Zhuang



Tommy Li



Andrew Butler



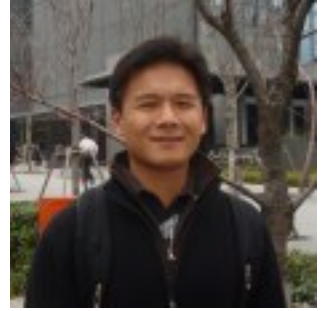
Jin Chi He



Feng Li



Ke Zhu



Kevin Yu





## Commits by Company

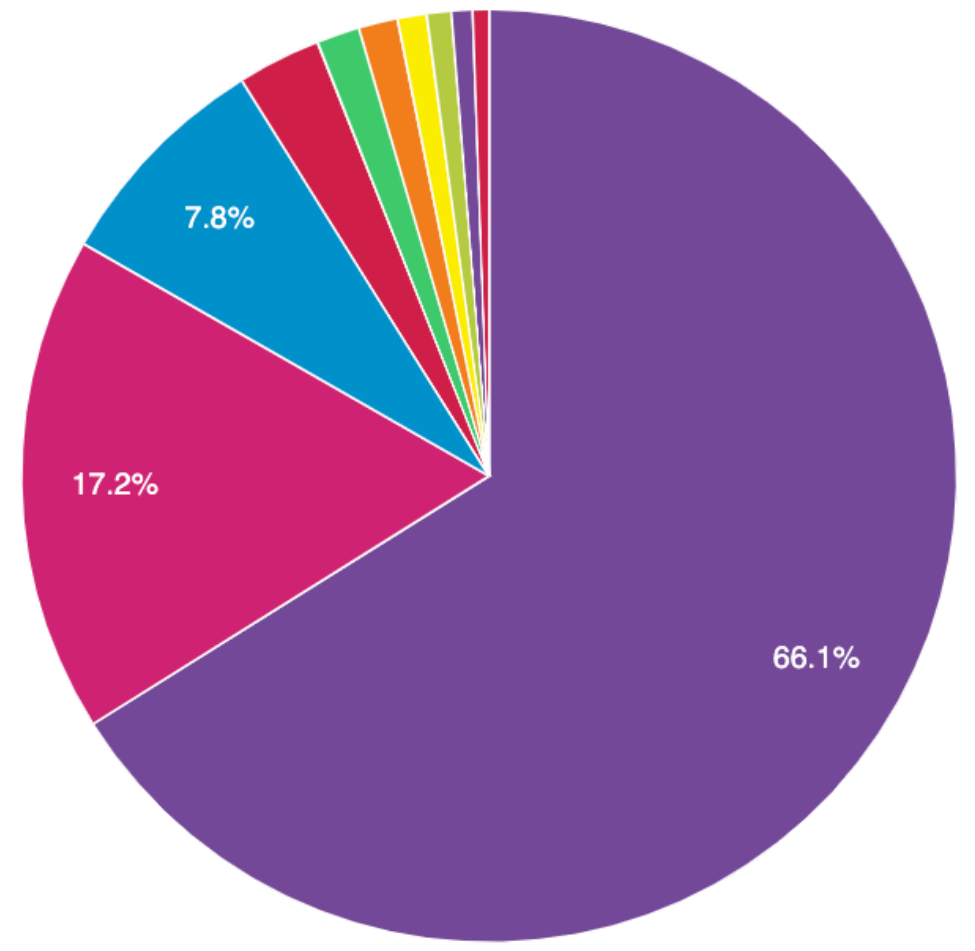
Show  entries

#	Company	Commits
	*independent	6882
1	Google	1792
2	IBM	816
3	Caicloud	301
4	Alibaba	141
5	Intel	105
6	Bloomberg LP	89
7	Red Hat	75
8	Huawei	59
9	Amazon	27

Showing 1 to 10 of 40 entries

[Previous](#)

[Next](#)





# IBM is the 2nd Largest Contributor



Kubeflow

Company	Contributions
Google	22064
IBM	4727
Cisco	4009
Caicloud	1865
Amazon	1425
Microsoft	553
Seldon	449
Net EASE	266
NetEase	260
Arrikto	213
DaoCloud	143
Huawei	139
NVidia	80
Oracle	78
Alibaba	70
Dell	63
Red Hat	52
Intel	50

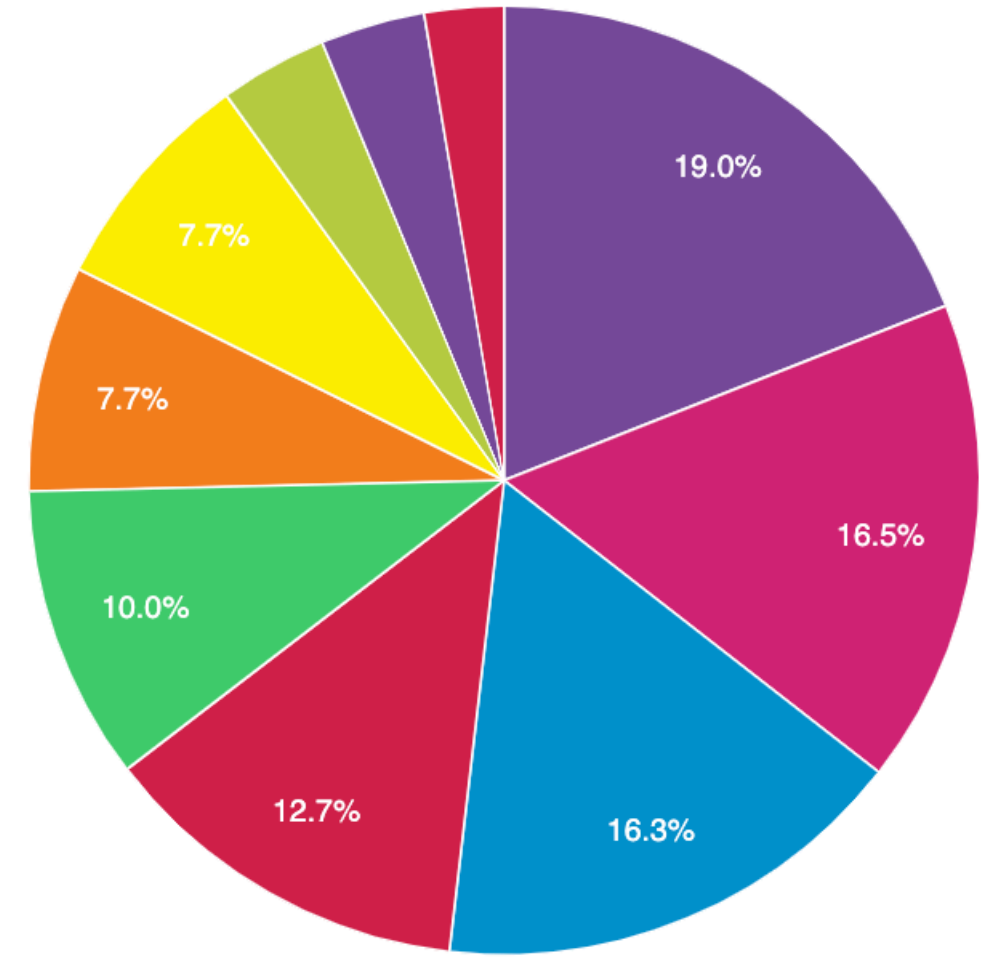


Show  entries

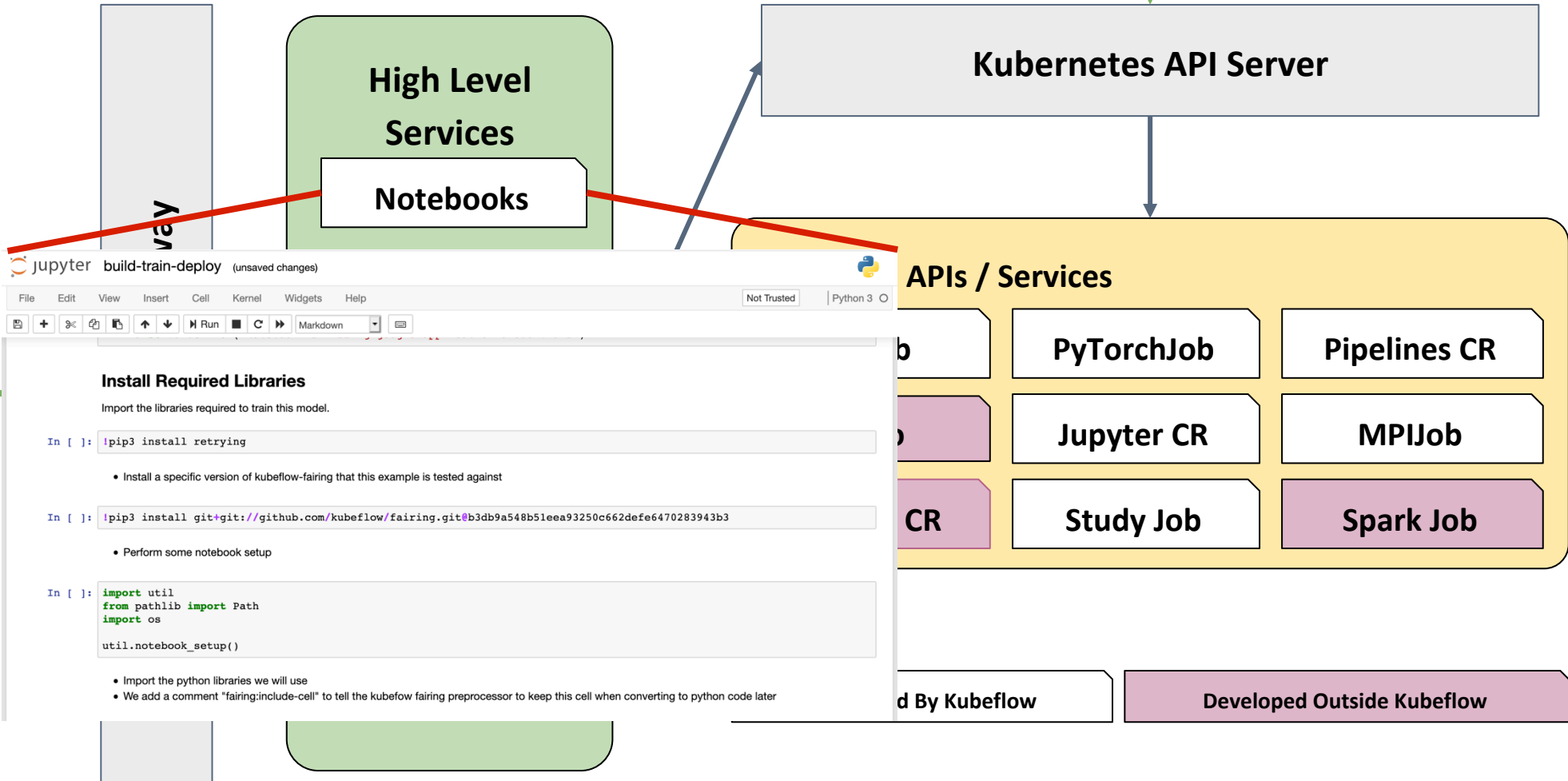
#	Module	Commits
1	kf-serving@kubeflow	155
2	kfp-tekton@kubeflow	135
3	katib@kubeflow	133
4	website@kubeflow	104
5	fairing@kubeflow	63
6	pipelines@kubeflow	63
7	examples@kubeflow	30
8	kfctl@kubeflow	29
9	kubeflow	22
10	manifests@kubeflow	21

Showing 1 to 10 of 18 entries

[Previous](#) [Next](#)



kubectl apply -f tfjob

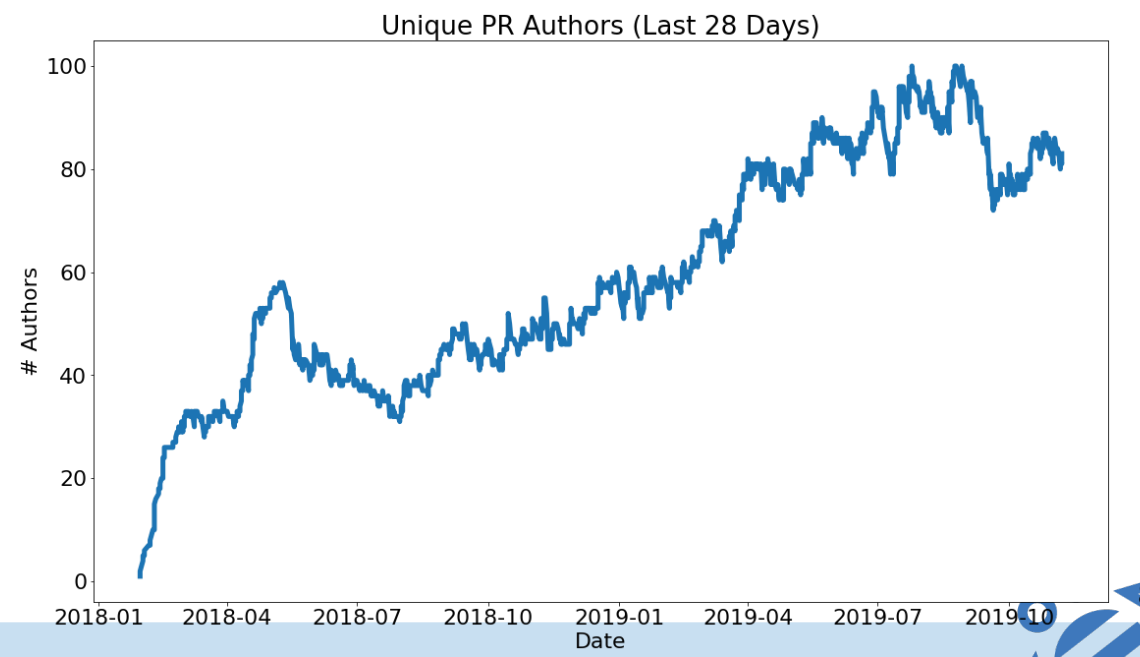
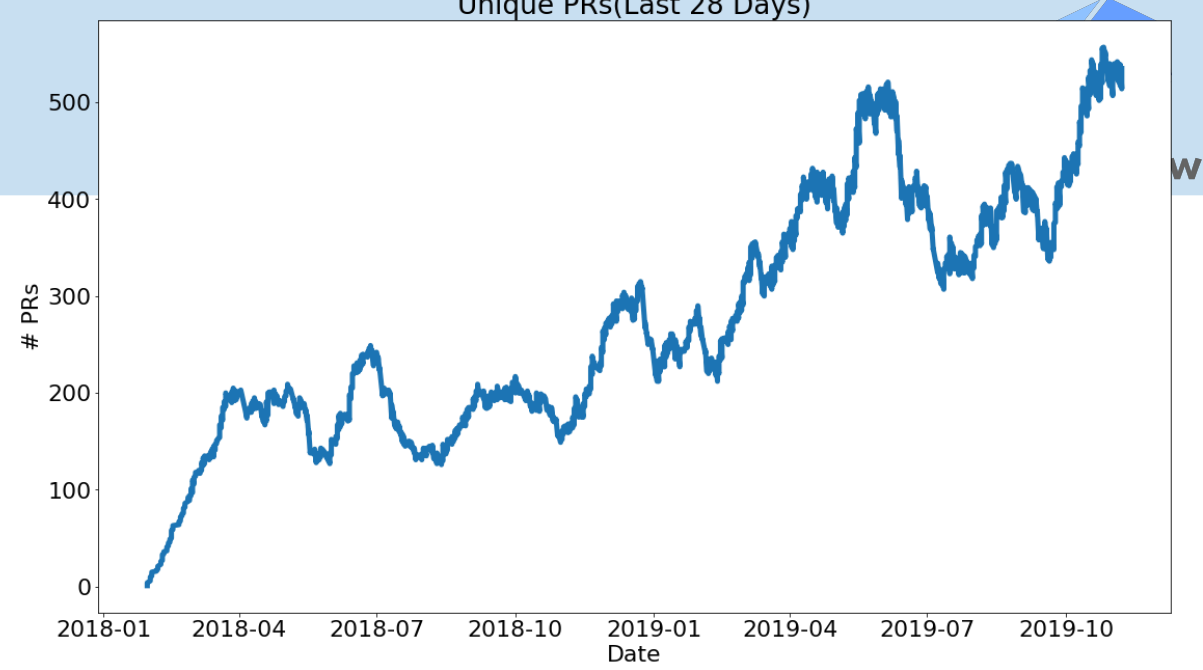


Adapted from Kubeflow Contributor Summit 2019 talk: Kubeflow and ML Landscape (Not all components are shown)



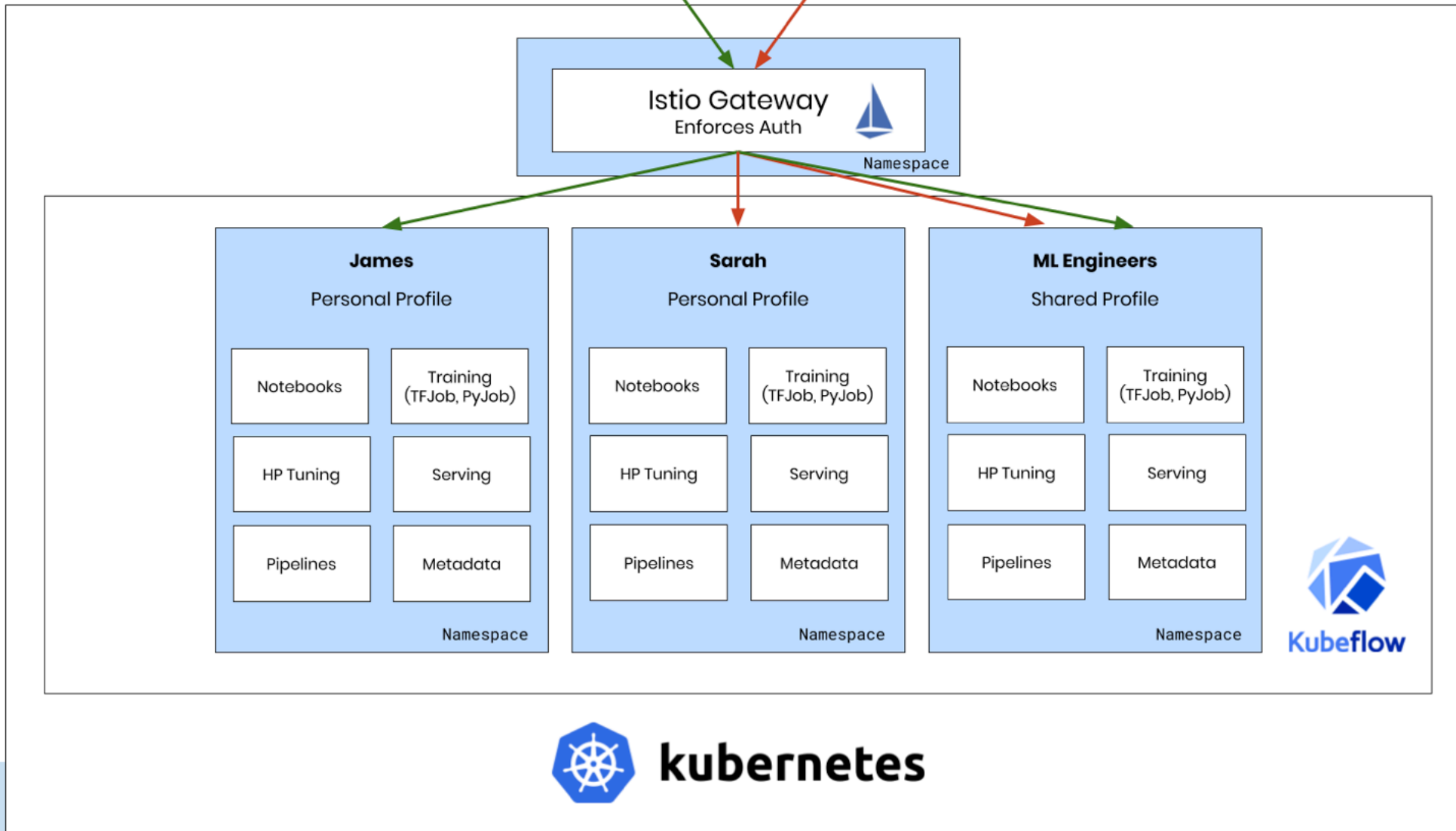


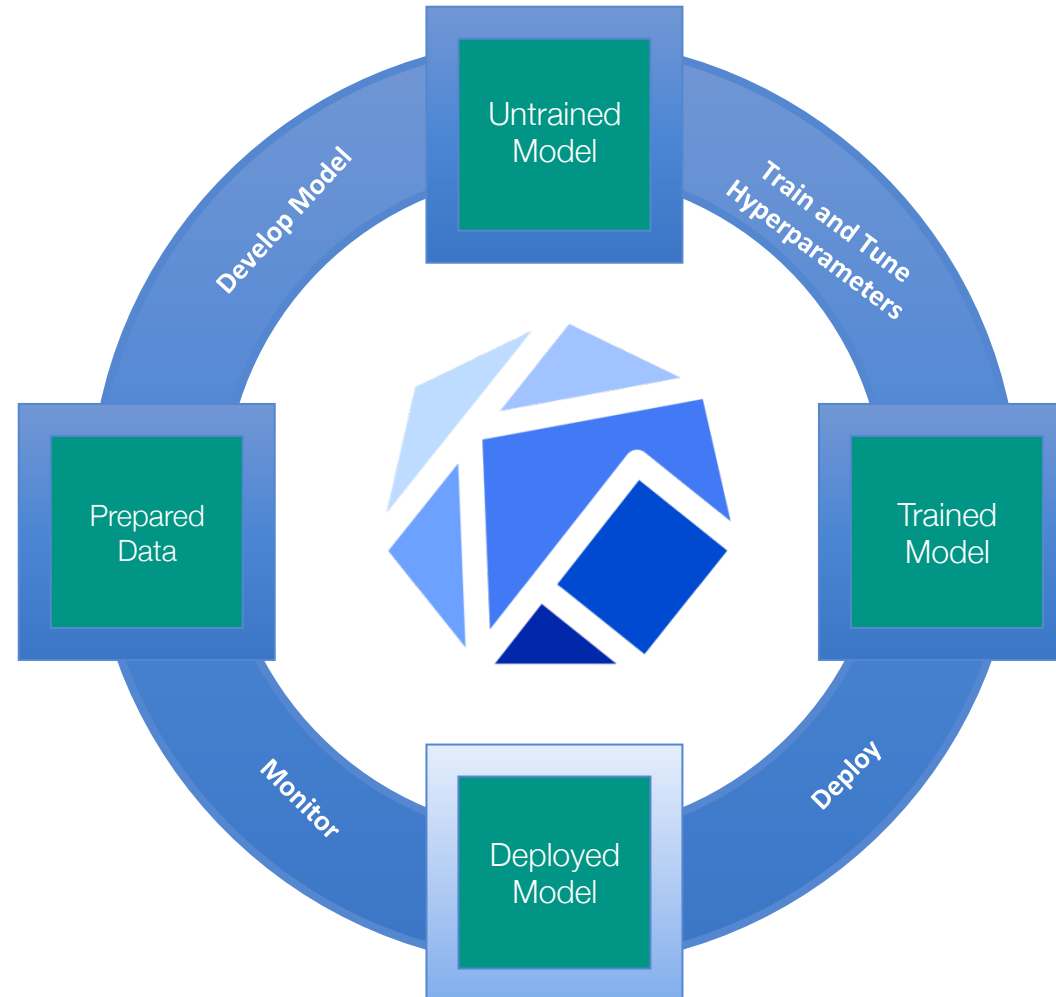
# Community is growing!





# Multi-User Isolation





# IBM Develop (Kubeflow Jupyter Notebooks)



Kubeflow

- 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





# Develop (Kubeflow Jupyter Notebooks)



Kubeflow

Kubeflow

Home

Pipelines

Notebook Servers

Katib

Artifact Store

---

GitHub

Documentation

Privacy • Usage Reporting

Select namespace ▼

Dashboard Activity

### Quick shortcuts

- Upload a pipeline**  
Pipelines
- View all pipeline runs**  
Pipelines
- Create a new Notebook server**  
Notebook Servers
- View Katib Studies**  
Katib
- View Metadata Artifacts**  
Artifact Store

### Recent Notebooks

*Choose a namespace to see Notebooks*

### Recent Pipelines

- refarch-reefer-ml**  
Created 6/29/2020, 10:04:11 AM
- [Tutorial] DSL - Control structures**  
Created 6/10/2020, 2:24:18 PM
- [Tutorial] Data passing in python components**  
Created 6/10/2020, 2:24:17 PM
- [Demo] TFX - Taxi Tip Prediction Model Trainer**  
Created 6/10/2020, 2:24:16 PM
- [Demo] XGBoost - Training with Confusion Matrix**  
Created 6/10/2020, 2:24:15 PM

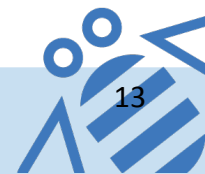
### Documentation

- Getting Started with Kubeflow**   
Get your machine-learning workflow up and running on Kubeflow
- MiniKF**   
A fast and easy way to deploy Kubeflow locally
- Microk8s for Kubeflow**   
Quickly get Kubeflow running locally on native hypervisors
- Minikube for Kubeflow**   
Quickly get Kubeflow running locally
- Kubeflow on GCP**   
Running Kubeflow on Kubernetes Engine and Google Cloud Platform
- Kubeflow on AWS**   
Running Kubeflow on Elastic Container Service and Amazon Web Services
- Requirements for Kubeflow**   
Get more detailed information about using Kubeflow and its components

### Recent Pipeline Runs



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



# Distributed Training Operators



**tf-operator**  
Tools for ML/Tensorflow on Kubernetes.  
● Jsonnet 🔒 Apache-2.0 🍴 323 ★

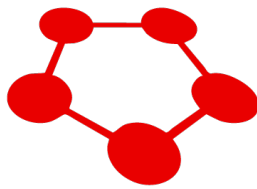
**pytorch-operator**  
PyTorch on Kubernetes  
● Jsonnet 🔒 Apache-2.0 🍴 87 ★ 11

**mpi-operator**  
Kubernetes Operator for Allreduce-style  
kubernetes tensorflow mpi dist  
horovod kubeflow  
● Go 🔒 Apache-2.0 🍴 83 ★ 125

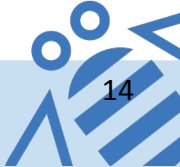
**xgboost-operator**  
Incubating project for xgboost operator  
● Go 🔒 Apache-2.0 🍴 23 ★ 41

**mxnet-operator**  
A Kubernetes operator for mxnet jobs  
● Go 🔒 Apache-2.0 🍴 20 ★ 50

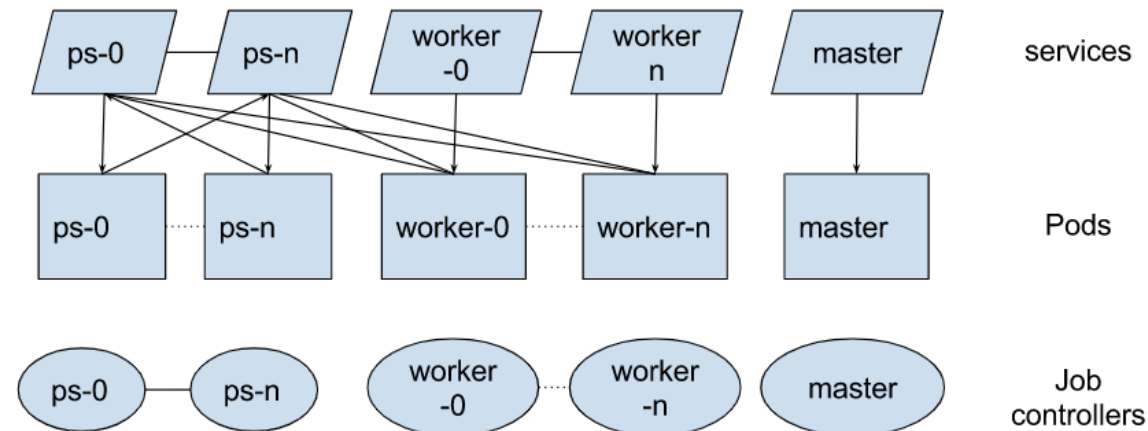
**chainer-operator**  
Repository for chainer operator  
● Go 🔒 Apache-2.0 🍴 9 ★ 12



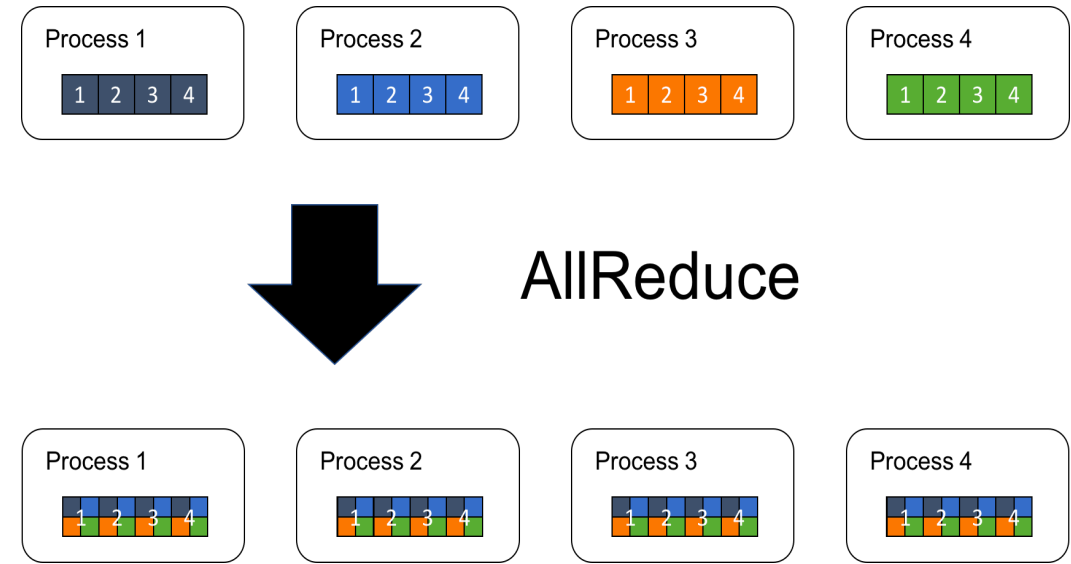
# Chainer



- 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



- 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





# IBM Hyper Parameter Optimization and Neural Architecture Search - Katib



- Katib: Kubernetes Native System for Automated tuning of machine learning model's Hyperparameter Tuning and Neural Architecture Search.
- Github Repository: [https://github.com/katib](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 \(DARTS\)](#)

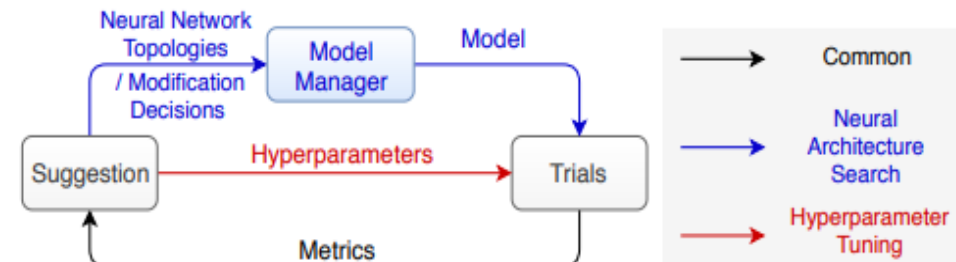
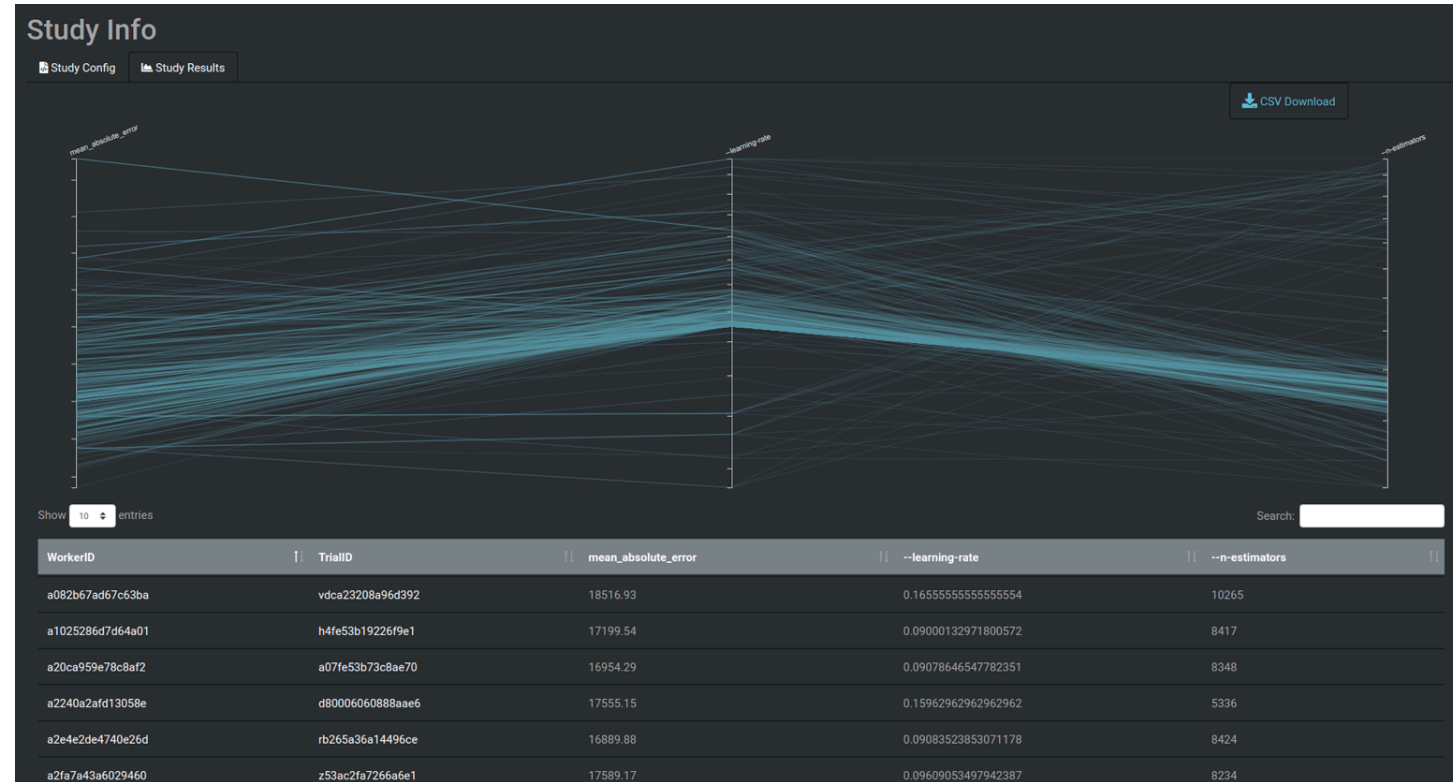


Figure 1: Summary of AutoML workflows



## Welcome to Katib

Choose type of experiment

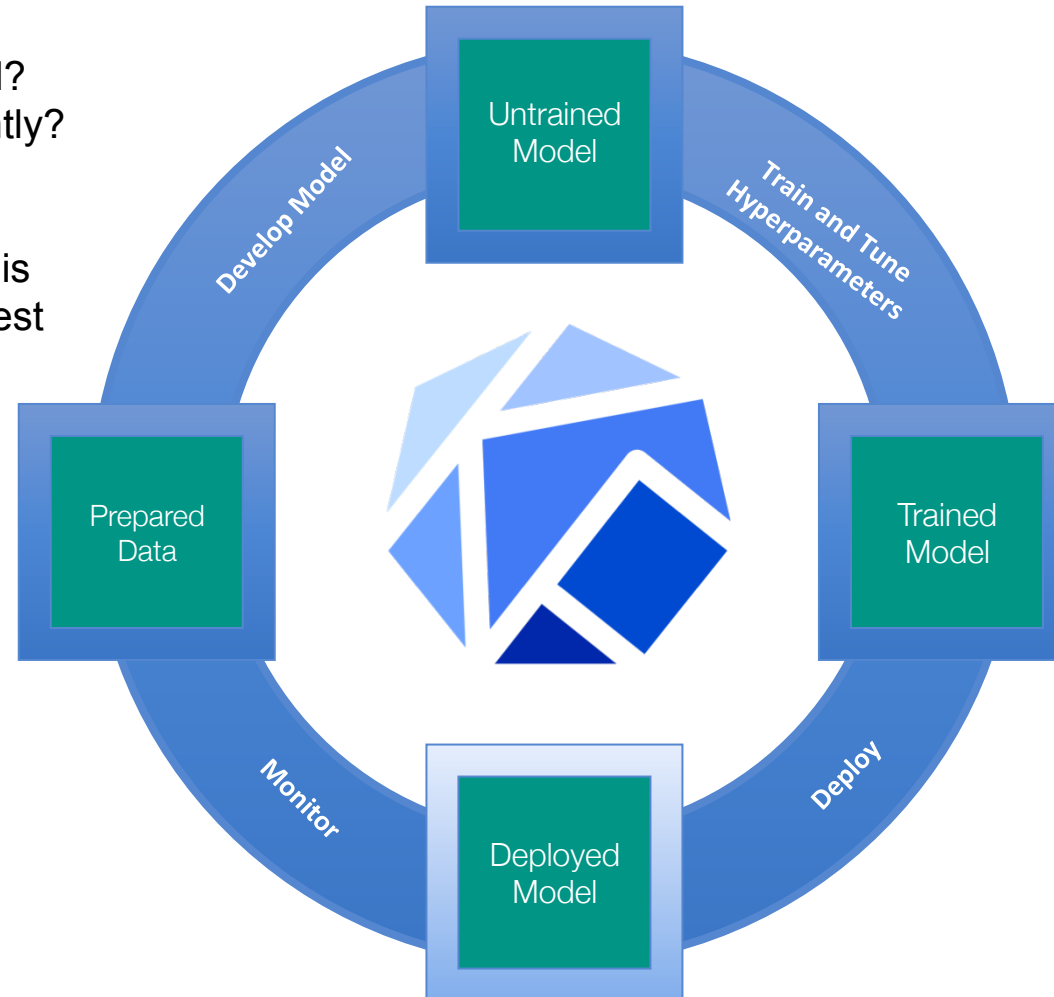
Hyperparameter  
Tuning

Neural Architecture  
Search

For usage instructions, see the [Kubeflow docs](#)

To contribute to Katib, visit [GitHub](#)

- 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 ML Serving 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



- 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 Kubernetesizing their Azure ML Stack



SELDON

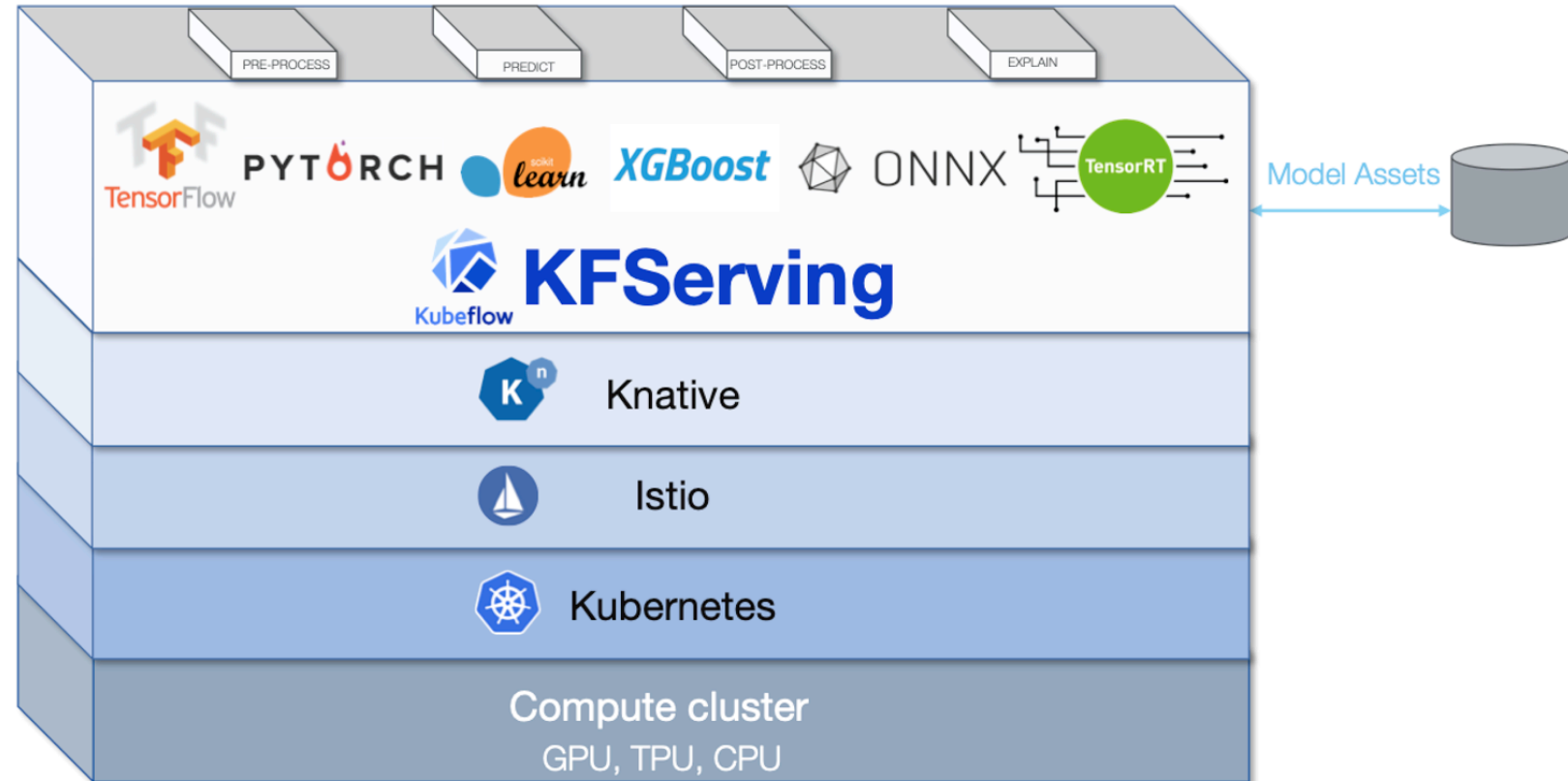
Bloomberg



- 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



- 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



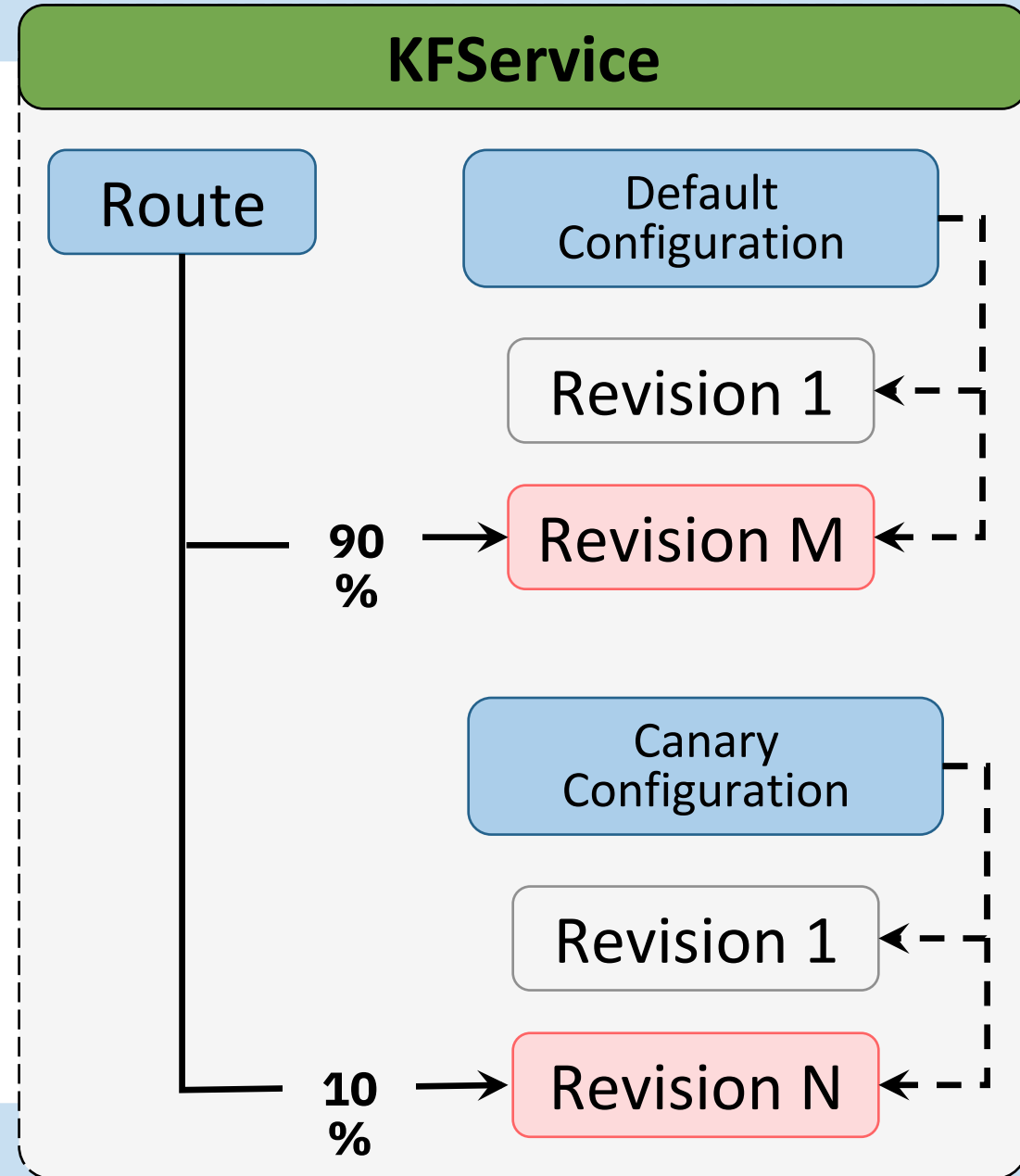


# KFServing: Default and Canary Configurations



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



## Model Servers

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

## Components:

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

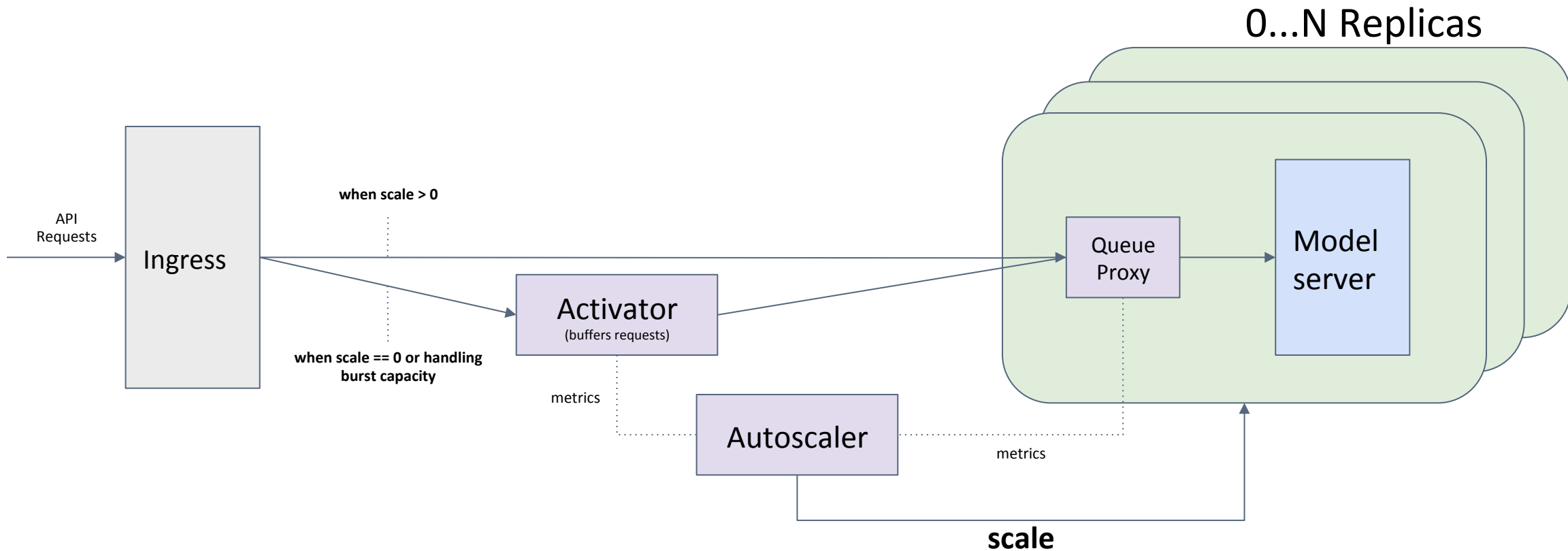
## Storage

- AWS/S3
- GCS
- Azure Blob
- PVC



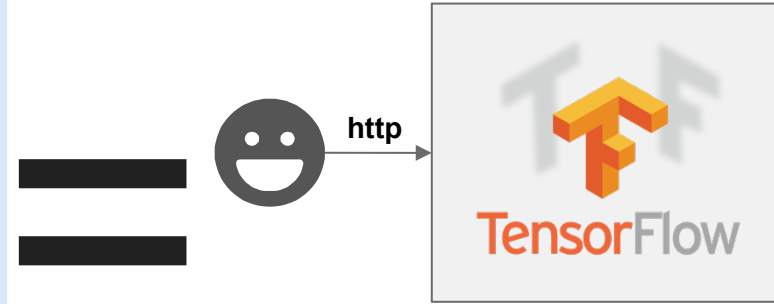


- Scale based on # in-flight requests against expected concurrency
- Simple solution for heterogeneous ML inference autoscaling





```
apiVersion: "serving.kubeflow.org/v1alpha2"
kind: "InferenceService"
metadata:
  name: "flowers-sample"
spec:
  default:
    predictor:
      tensorflow:
        storageUri: "gs://kfserving-samples/models/tensorflow/flowers"
```



- A pointer to a Serialized Model File
- 9 lines of YAML
- A live model at an HTTP endpoint

- Scale to Zero
- GPU Autoscaling
- Safe Rollouts
- Optimized Serving Containers
- Network Policy and Auth
- HTTP APIs (gRPC soon)
- Tracing
- Metrics

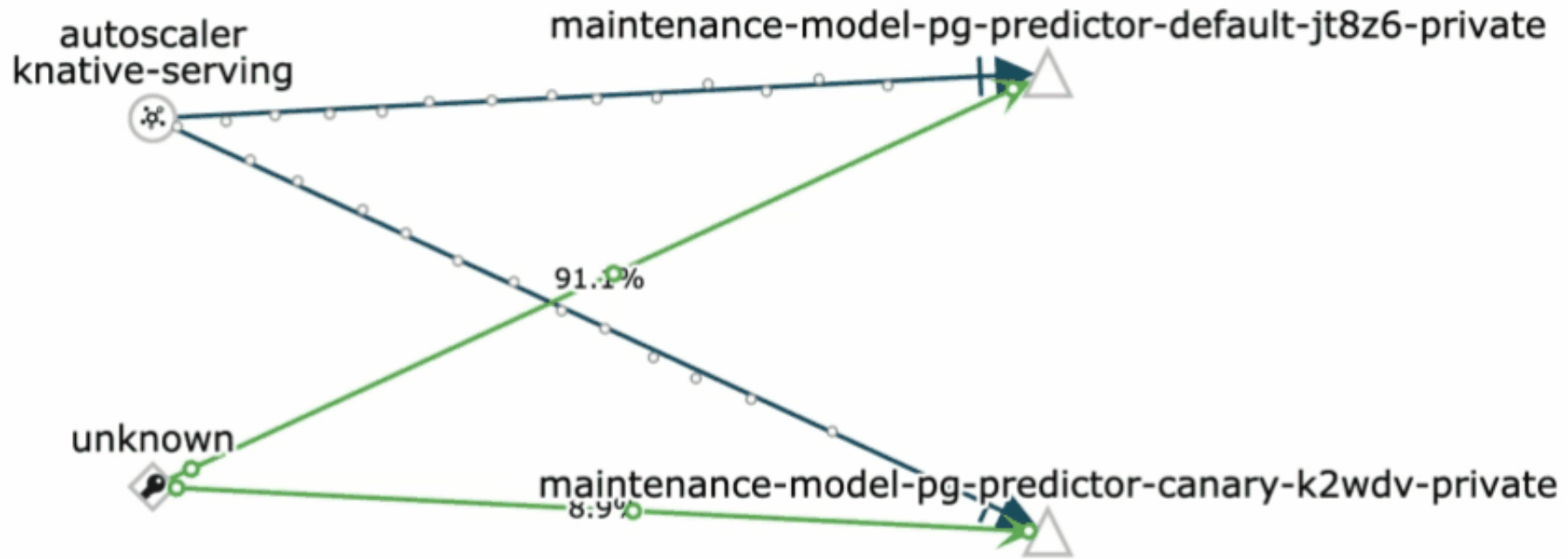
Production users include:

**Bloomberg**



**gojek**





Hide

Legend

0s 889 x 537

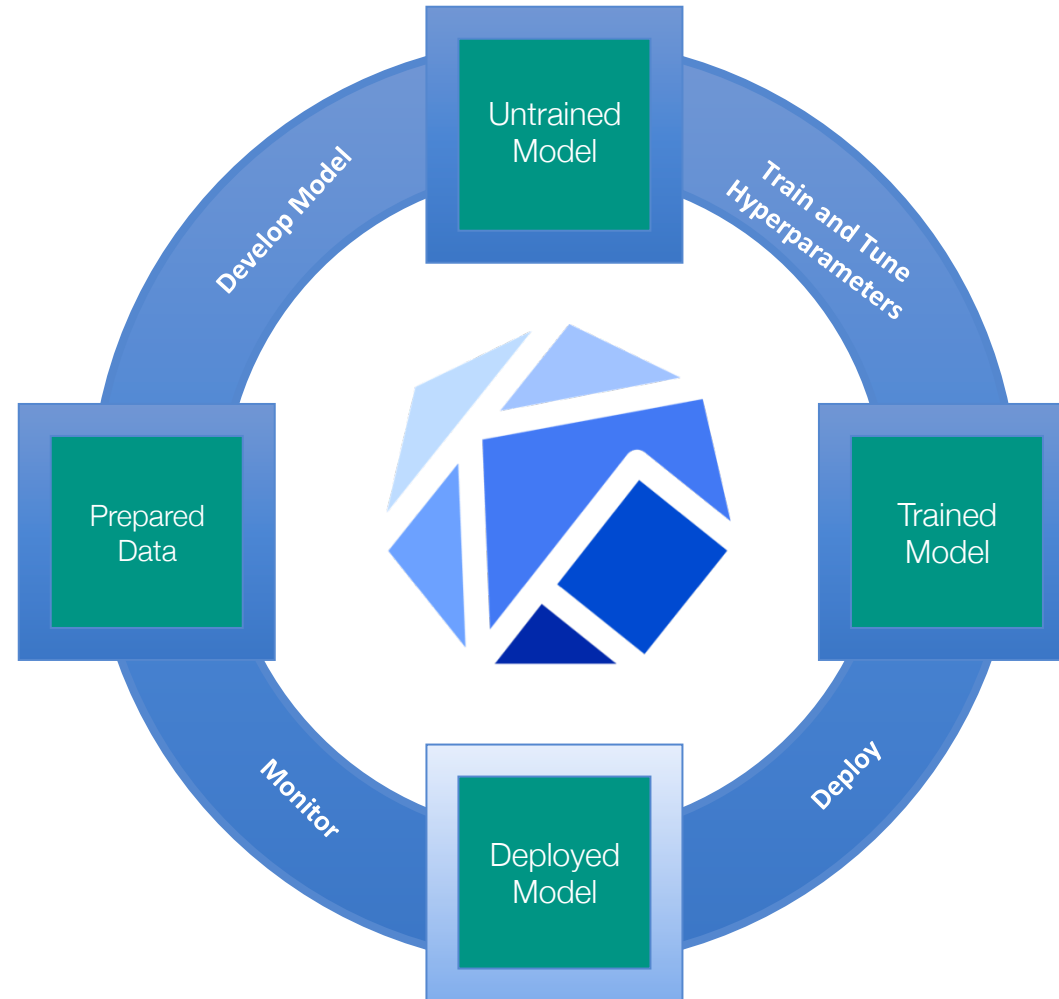


- ❑ Crowd sourced capabilities – Contributions by AWS, Bloomberg, Google, Seldon, IBM, NVidia and others.
- ❑ Support for multiple runtimes pre-integrated (TFServing, Nvidia Triton (GPU optimization), ONNX Runtime, SKLearn, PyTorch, XGBoost, Custom models).
- ❑ Serverless ML Inference and Autoscaling: Scale to zero (with no incoming traffic) and Request queue based autoscaling
- ❑ Canary and Pinned rollouts: Control traffic percentage and direction, pinned rollouts
- ❑ Pluggable pre-processor/post-processor via Transformer: Gives capabilities to plug in pre-processing/post-processing implementation, control routing and placement (e.g. pre-processor on CPU, predictor on GPU)
- ❑ Pluggable analysis algorithms: Explainability, Drift Detection, Anomaly Detection, Adversarial Detection (contributed by Seldon) enabled by Payload Logging (built using CloudEvents standardized eventing protocol)
- ❑ Batch Predictions: Batch prediction support for ML frameworks (TensorFlow, PyTorch, ...)
- ❑ Integration with existing monitoring stack around Knative/Istio ecosystem: Kiali (Service placements, traffic and graphs), Jaeger (request tracing), Grafana/Prometheus plug-ins for Knative)
- ❑ Multiple clients: kubectl, Python SDK, Kubeflow Pipelines SDK
- ❑ Standardized Data Plane V2 protocol for prediction/explainability et al: Already implemented by Nvidia Triton



- ❑ MMS: Multi-Model-Serving for serving multiple models per custom KFService instance
- ❑ More Data Plane v2 API Compliant Servers: SKLearn, XGBoost, PyTorch...
- ❑ Multi-Model-Graphs and Pipelines: Support chaining multiple models together in a Pipelines
- ❑ PyTorch support via AWS TorchServe
- ❑ gRPC Support for all Model Servers
- ❑ Support for multi-armed-bandits
- ❑ Integration with IBM AIX360 for Explainability, AIF360 for Bias detection and ART for Adversarial detection





- 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

The screenshot displays the Kubeflow Pipelines interface. At the top, there are navigation buttons: '+ Create run' (highlighted with a red circle), 'Upload version', '+ Create experiment', and 'Delete'. Below this is a breadcrumb trail: '[Demo] TFX - Taxi Tip Prediction Model Trainer ([Demo] TFX - T...'. The main area shows a 'Graph' view of a pipeline with the following steps: 'csvexamplegen' (root), 'statisticsgen', 'schemagen', 'examplevalidator', 'transform', 'trainer', 'evaluator', 'modelvalidator', and 'pusher'. The 'evaluator' and 'modelvalidator' steps are connected to the 'csvexamplegen' step. The 'transform' and 'trainer' steps are connected to 'statisticsgen' and 'schemagen'. The 'pusher' step is connected to 'modelvalidator' and 'trainer'.

At the bottom, a 'Pipelines' table is visible, listing various pipeline samples:

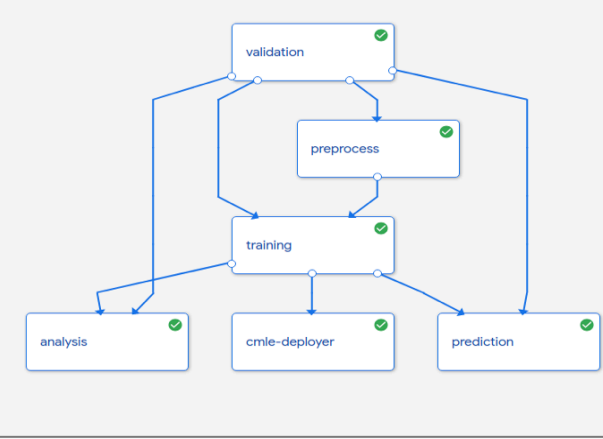
<input type="checkbox"/>	Pipeline name	Description	Uploaded on ↓
<input type="checkbox"/>	[Sample] Basic - Condition	A pipeline shows how to use dsl.Condition. For source code, refer to https://github.com/ku...	02/01/2019, 11:24:37
<input type="checkbox"/>	[Sample] Basic - Exit Handler	A pipeline that downloads a message and print it out. Exit Handler will run at the end. For s...	02/01/2019, 11:24:36
<input type="checkbox"/>	[Sample] Basic - Immediate ...	A pipeline with parameter values hard coded. For source code, refer to https://github.com/ku...	02/01/2019, 11:24:34
<input type="checkbox"/>	[Sample] Basic - Parallel Join	A pipeline that downloads two messages in parallel and print the concatenated result. For ...	02/01/2019, 11:24:33
<input type="checkbox"/>	[Sample] Basic - Sequential	A pipeline with two sequential steps. For source code, refer to https://github.com/kubeflo...	02/01/2019, 11:24:32
<input type="checkbox"/>	[Sample] ML - TFX - Taxi Tip ...	Example pipeline that does classification with model analysis based on a public tax cab BL...	02/01/2019, 11:24:30
<input type="checkbox"/>	[Sample] ML - XGBoost - Trai...	A trainer that does end-to-end distributed training for XGBoost models. For source code, re...	02/01/2019, 11:24:29



# 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

The screenshot displays the Kubeflow dashboard interface. On the left is a navigation sidebar with options: Pipelines, Experiments, Artifacts, Executions, Archive, Documentation, Github Repo, and AI Hub Samples. The main area shows a pipeline execution graph with nodes: csvexampleger, statisticsgen, schemagen, examplevalidator, resolvernode-lates..., evaluator, train, and pusher. The 'evaluator' node is highlighted with a green checkmark. A modal window titled 'Artifacts' is open, showing a 'Static HTML' artifact viewer. The viewer includes a 'Sort by' dropdown set to 'Feature', a 'Reverse order' checkbox, and a 'Feature search' field. Below these are feature selection checkboxes for 'int(8)', 'float(7)', 'string(2)', and 'unknown(1)'. A table titled 'Numeric Features (15)' displays the following data:

	count	missing	mean	std dev
dropoff_census_tract	3,618	28.93%	17.0B	333k
dropoff_community_area	4,905	3.65%	21.2	17.85
dropoff_latitude	4,915	3.46%	41.9	0.04
dropoff_longitude	4,915	3.46%	-87.65	0.06



# IBM Pipelines versioning



Kubeflow

## Pipelines

+ Upload pipeline

Refresh

Delete

Filter pipelines



<input type="checkbox"/>	Pipeline name	Description	Uploaded on ↓
<input type="checkbox"/>	▶ [Tutorial] DSL - Control structures	<a href="#">source code</a> Shows how to use conditional execution and exit handlers. This pipeline will randomly fail to demonstra...	2/20/2020, 3:28:12 PM
<input type="checkbox"/>	▶ [Tutorial] Data passing in python com...	<a href="#">source code</a> Shows how to pass data between python components.	2/20/2020, 3:28:11 PM
<input type="checkbox"/>	▼ [Demo] TFX - Taxi Tip Prediction Mod...	<a href="#">source code</a> <a href="#">GCP Permission requirements</a> . Example pipeline that does classification with model analysis based on ...	2/20/2020, 3:28:10 PM
<input type="checkbox"/>	Version name		Uploaded on ↓
<input type="checkbox"/>	TFX - Taxi Tip Prediction Model Trainer_version_at_2020-03-03T15:44:30.197Z		3/3/2020, 7:55:03 AM
<input type="checkbox"/>	[Demo] TFX - Taxi Tip Prediction Model Trainer		2/20/2020, 3:28:10 PM
<input type="checkbox"/>	▶ [Demo] XGBoost - Training with Confu...	<a href="#">source code</a> <a href="#">GCP Permission requirements</a> . A trainer that does end-to-end distributed training for XGBoost models.	2/20/2020, 3:28:09 PM

Rows per page: 10 < >

Rows per page: 10 < >

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





Getting Started

Pipelines

Experiments

**Artifacts**

Executions

Archive

Documentation

Github Repo

AI Hub Samples

### Artifacts

Filter

Pipeline/Workspace	Name	ID	Type	URI	Created at
		1	ExternalArtifact	<a href="#">gs://ml-pipeline-playground/tfx_t...</a>	
taxi_pipeline_with_parameters	examples	2	Examples	<a href="#">gs://aju-pipelines/tfx_taxi_simpl...</a>	2/20/2020, 5:1...
	statistics	3	ExampleStatistics	<a href="#">gs://aju-pipelines/tfx_taxi_simpl...</a>	2/20/2020, 5:1...
	schema	4	Schema	<a href="#">gs://aju-pipelines/tfx_taxi_simpl...</a>	2/20/2020, 5:1...
	anomalies	5	ExampleAnomalies	<a href="#">gs://aju-pipelines/tfx_taxi_simpl...</a>	2/20/2020, 5:1...
	transform_graph	6	TransformGraph	<a href="#">gs://aju-pipelines/tfx_taxi_simpl...</a>	2/20/2020, 5:1...
	transformed_e...	7	Examples	<a href="#">gs://aju-pipelines/tfx_taxi_simpl...</a>	2/20/2020, 5:1...
	model	8	Model	<a href="#">gs://aju-pipelines/tfx_taxi_simpl...</a>	2/20/2020, 5:2...
	evaluation	9	ModelEvaluation	<a href="#">gs://aju-pipelines/tfx_taxi_simpl...</a>	2/20/2020, 5:2...
	blessing	10	ModelBlessing	<a href="#">gs://aju-pipelines/tfx_taxi_simpl...</a>	2/20/2020, 5:2...
	pushed_model	11	PushedModel	<a href="#">gs://aju-pipelines/tfx_taxi_simpl...</a>	2/20/2020, 5:2...
	evaluation	12	ModelEvaluation	<a href="#">gs://aju-pipelines/tfx_taxi_simpl...</a>	2/20/2020, 5:4...

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.

Artifacts

← model

Overview Lineage Explorer

**Type: Model**

URI  
[gs://aju-pipelines/tfx\\_taxi\\_simple/85265540-6a06-4969-a49e-1f65741878be/Trainer/model/7](#)

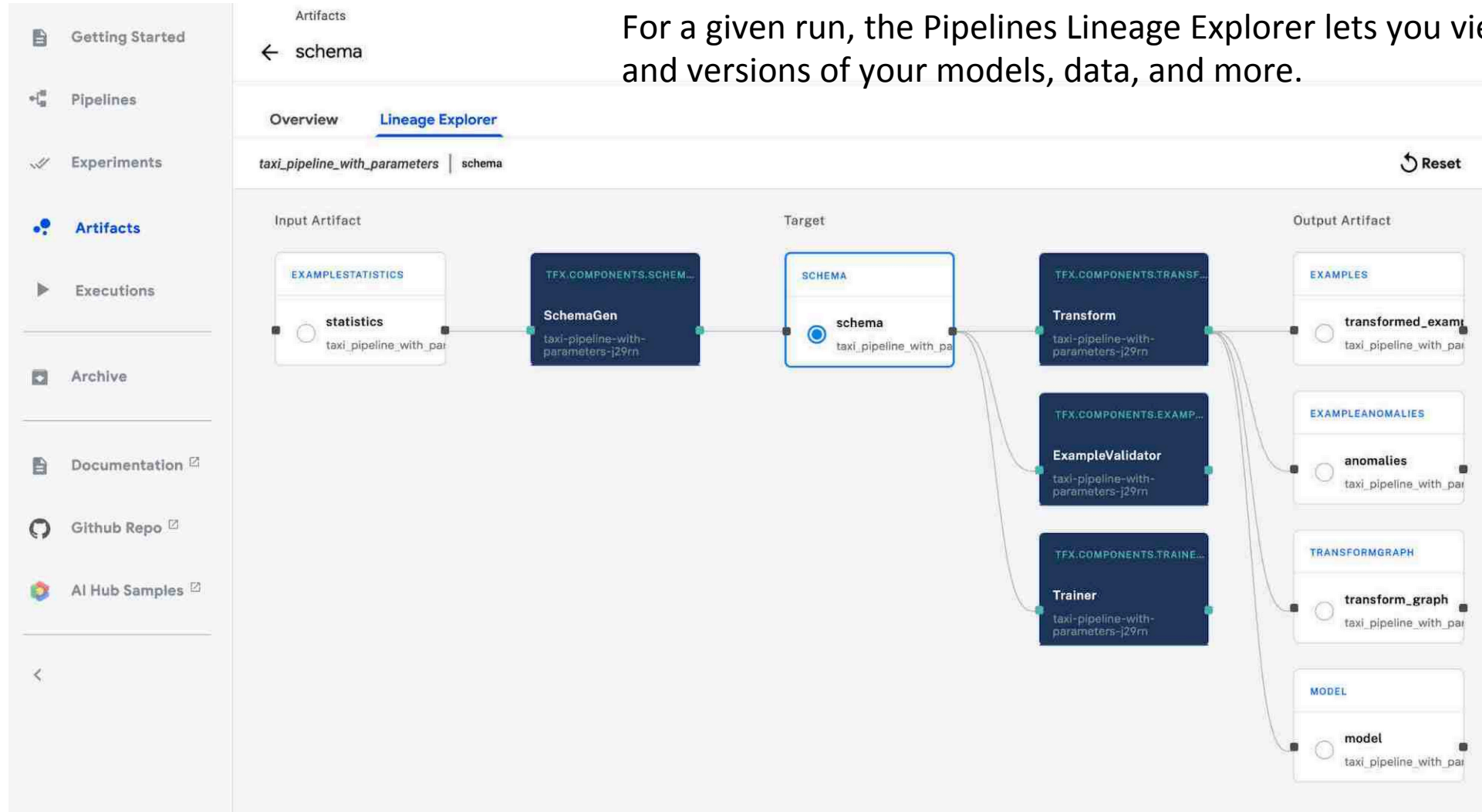
**Properties**

**Custom Properties**

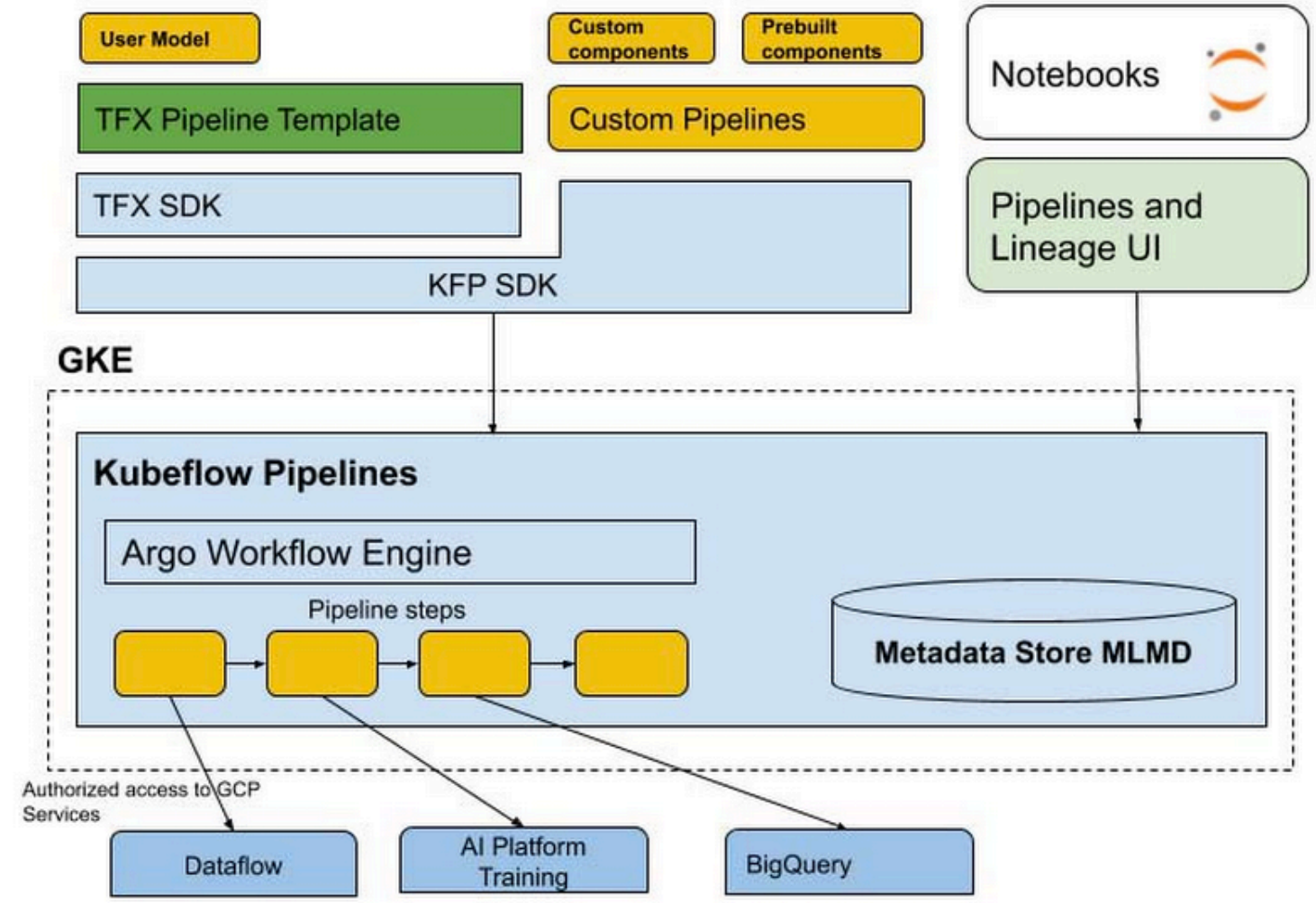
name	pipeline_name	producer_component	state
model	taxi_pipeline_with_parameters	Trainer	published



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



# Kubeflow Pipeline Architecture



# Kubeflow Pipelines can train, deploy and serve

Experiments > KFServing Experiments

← animesh-refarch-reefer-ml (f6766) Retry Clone run Terminate Archive

**Graph** Run output Config

```
graph TD; training[training];
```

icp4d-demo-xgngg-3720630081

Artifacts Input/Output Volumes Manifest **Logs**

```
1 - Initializing github client
2 - Initializing object storage client
3 - Downloading notebook: https://raw.githubusercontent.com/Tomcli/notebooks/master/notebooks/sklearn-pg.ipynb
4 - Download successful
5 - Parsing notebook parameters
6 - Parameter parsing successful
7 - Executing notebook: sklearn-pg.ipynb
8 - Notebook Parameters: {}
9
```

Runtime execution graph. Only steps that are currently running or have already completed are shown.

0s | 1199 x 669

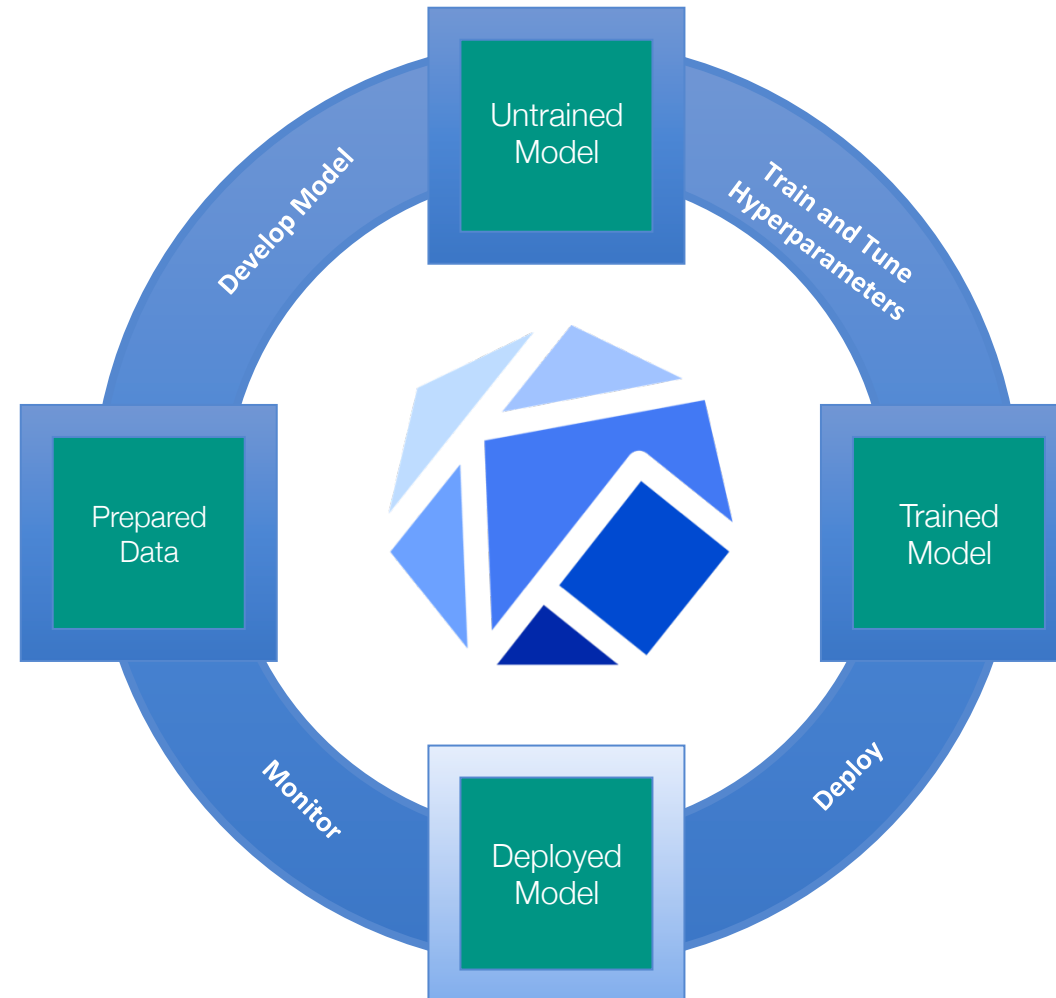




Kubernetes Ready



**ML and AI Platform**



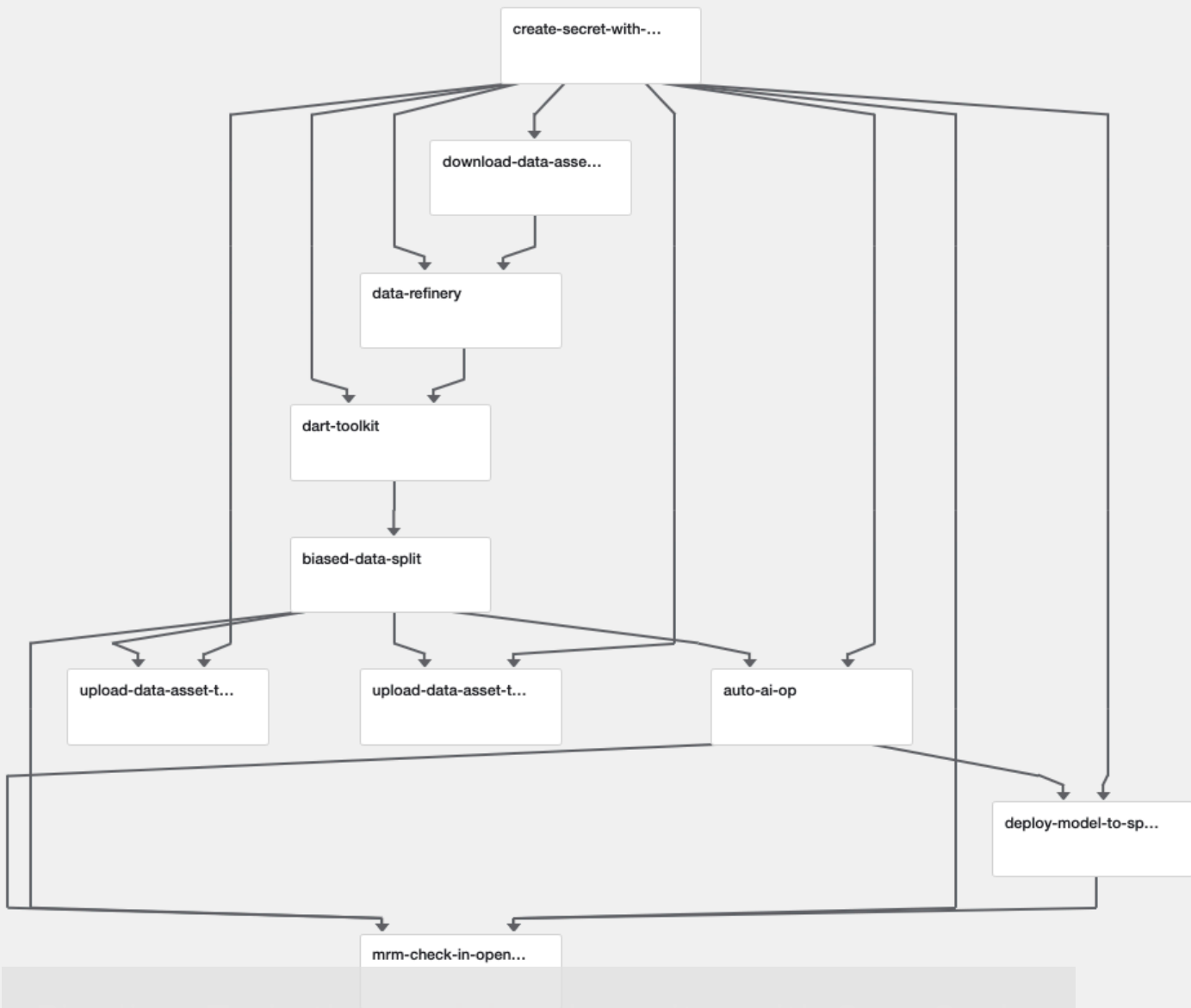


# Watson AI Pipelines

- Demonstrate that Watson can be used for end-end AI lifecycle data prep/model training/model risk validation/model deployment/monitoring/updating models
- Demonstrate that the full lifecycle can be operated programmatically, and have **Tekton** as a backend instead of Argo

```
19 Requirement already satisfied: numpy in /opt/app-root/1/lib/ovh/3.6/kite-packages (from line=>0.1.1.33) (1
20 Requirement already satisfied: joblib=>0.11 in
21 Requirement already satisfied: python-dateutil>
22 Requirement already satisfied: pyprindl=>2.0.4
23 Requirement already satisfied: six=>1.10 in /op
24 Requirement already satisfied: pytz in /opt/app
25 Requirement already satisfied: cyclot=>0.10 in
26 Requirement already satisfied: PyWavelets=>0.4
27 Requirement already satisfied: networkx=>2.0 in
28 Requirement already satisfied: pillow=>3.0.0
29 Requirement already satisfied: imageio=>2.3.0
30 Requirement already satisfied: decorator=>4.3.0
31 - drift...
32 enablement finished.
33 running nrm...
34 [{"entity": {"data_mart_id": "00000000-0000-000
35 MM monitor instance: 4b451ccc-5dfb-42af-9926-0
36 Triggering MM computation with https://rcs.co
37 Done triggering MM computation with nrm_monit
38 running upload and evaluate for validation_test
46 running upload and evaluate for validation_test
47 10:40:56 upload_in_progress
48 10:41:12 running
49 10:41:27 running
50 10:41:43 running
51 10:41:55 finished
52 running upload and evaluate for validation_test
53 10:42:01 upload_in_progress
54 10:42:14 upload_in_progress
55 10:42:28 running
56 10:42:44 running
57 10:42:58 running
58 10:43:11 finished
59 running upload and evaluate for validation_test
60 10:43:17 upload_in_progress
61 10:43:29 upload_in_progress
62 10:43:44 running
63 10:43:59 running
64 10:44:11 running
65 10:44:25 running
66 10:44:37 finished
67 running upload and evaluate for validation_test
68 10:44:43 upload_in_progress
69 10:44:55 upload_in_progress
70 10:45:08 running
71 10:45:21 running
72 10:45:35 running
73 10:45:48 finished
74
```

Rank	Name	Algorithm	Accuracy (Optimized)	Enhancements	Build time
1	Pipeline 4	Gradient Boosting Classifier	0.807	HPO-1 FE HPO-2	00:01:48
2	Pipeline 3	Gradient Boosting Classifier	0.804	HPO-1 FE	00:04:19
3	Pipeline 2	Gradient Boosting Classifier	0.804	HPO-1	00:00:38
4	Pipeline 1	Gradient Boosting Classifier	0.802	None	00:00:07



### Run details

Pipeline \*  
 Train the model and monitor with OpenScale Choose

Pipeline Version \*  
 Train the model and monitor with OpenScale Choose

Run name \*  
 Run of Train the model and monitor with OpenScale (a28a6)

Description (optional)

This run will be associated with the following experiment

Experiment \*  
 GCR-AutoAI-Experiment-1 Choose

Run Type  
 One-off  Recurring

Run parameters  
 Specify parameters required by the pipeline

github\_token  
 6fd86cff0394892e772cd84d43a9e2d7546b1576

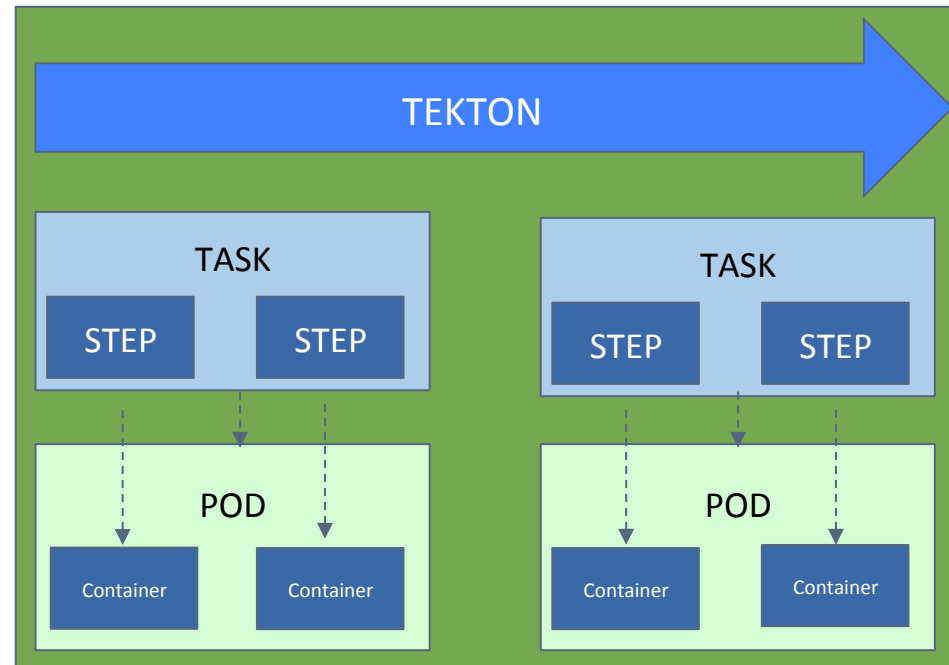
ai\_config\_url  
 https://raw.githubusercontent.com/Al-Lifecycle-Poland/kubeflow-pipelines-credentials/master/config\_cpd

catalog\_name  
 DataCatalog

asset\_id  
 2737bafc-3f78-4e2d-850a-e7f352b3d6b8

pre\_production\_space\_uid  
 1dd2aaec-781a-4712-a7ff-ae1862cf7a84

- ❑ 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.

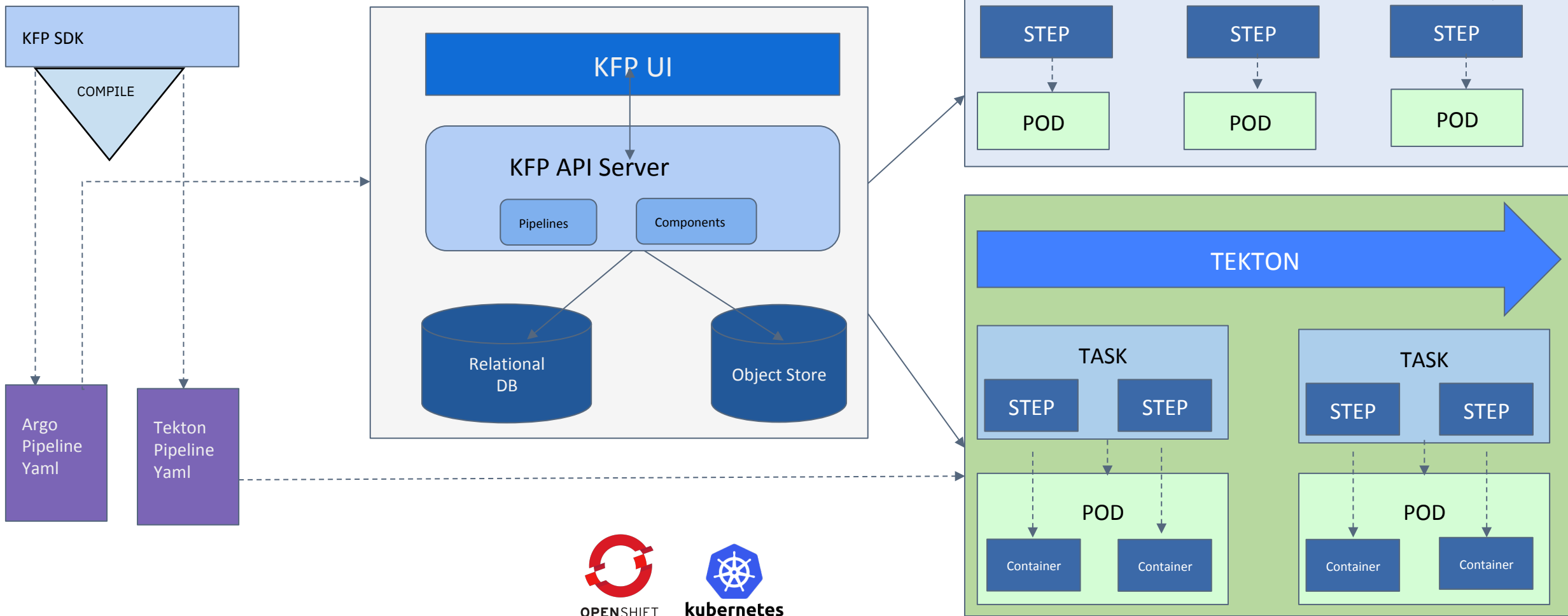


- ❑ A **PipelineResource** defines an object that is an input (such as a git repository) or an output (such as a docker image) of the pipeline.
- ❑ A **PipelineRun** defines an execution of a pipeline. It references the Pipeline to run and the PipelineResources to use as inputs and outputs.
- ❑ A **Pipeline** 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



OPENSIFT



kubernetes



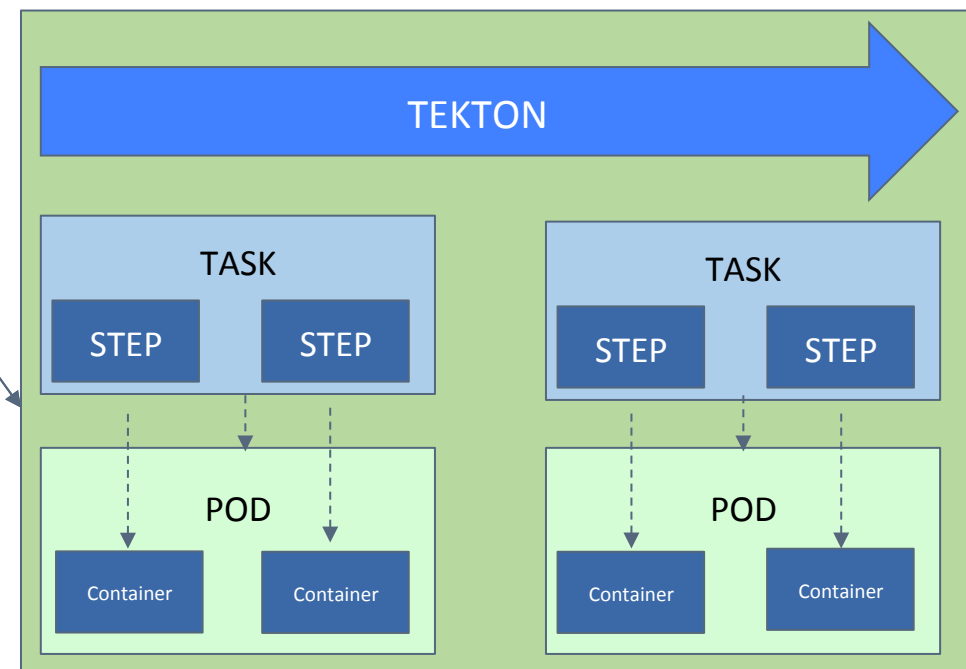
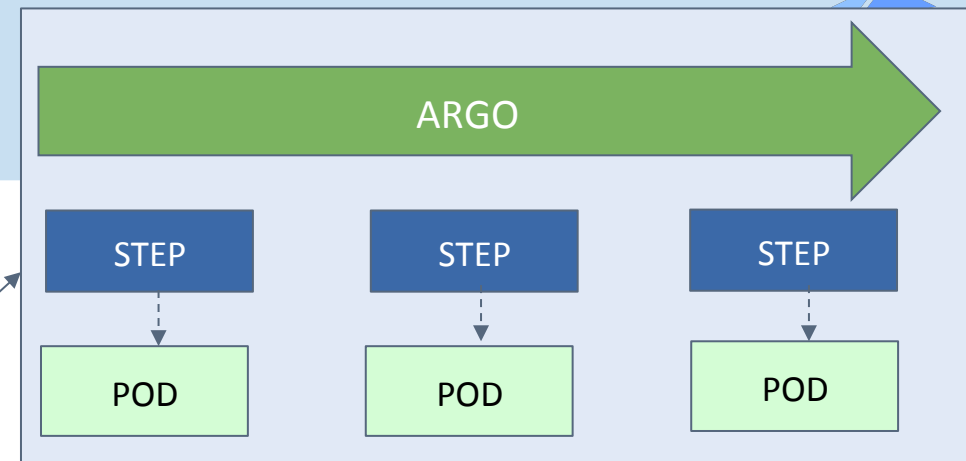
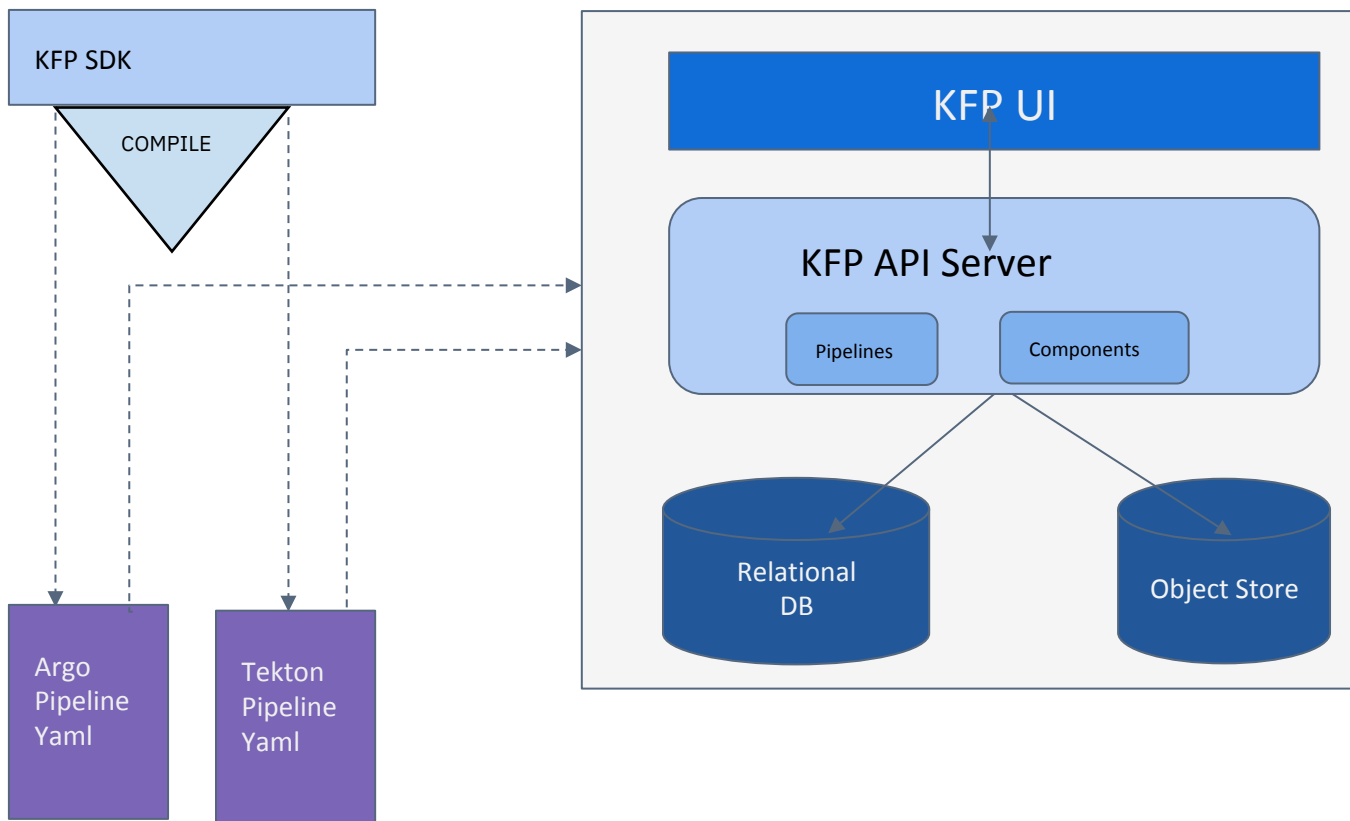
## Pluggable Components

- 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
- ...
- ...



## Multiple Moving parts, with different stakeholders

**Tekton Community:** Argo with version 2.6 much more mature than Tekton v0.11 (alpha) when the work started around 5 months ago

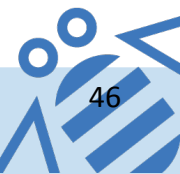
- Multiple features and capabilities lacking in Tekton when we kick started
- The team had to default to a spreadsheet to start tracking and mapping KFP DSL features, and areas where Tekton needed to bring features and functions.

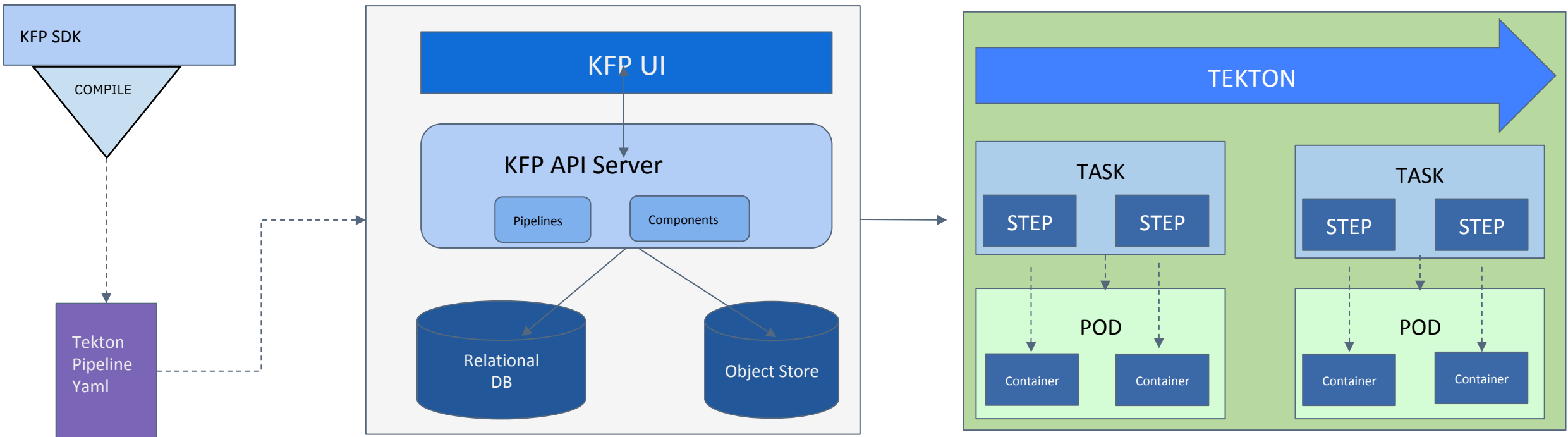
Overall 50 DSL capabilities identified and corresponding Tekton features started getting mapped.

- Multiple features like Kubernetes resources support to create/patch/update/delete them, image pull secrets, loops, conditionals, support for system params didn't exist. Or existed partially
- Tekton started moving from alpha to beta as the work progressed, and few features left behind in alpha mode
- Multiple issues opened on Tekton. Required ramping up the team of Tekton contributors to help drive these issues . Formed a virtual team of IBM Open tech developers (Andrea Frittoli, Priti Desai), IBM Systems team (Vincent Pli) DevOps team (Simon Kaegi), RedHat (Vincent Demeester etc.) to drive Tekton requirements

**Kubeflow Pipeline and TFX Community:** Open source team needed to be formed for the specific mission. And trained. Additionally Google needed to be brought up on the same page, and convinced the validity of integration.

- Multiple design reviews established with Google, and jointly agreed on a direction after they were convinced why we were doing it, and why it makes sense.
- Convincing to accelerate the IR (Intermediate Representation) strategy with TFX, so as to be able to drive this the right way
- Huge dependency in Kubeflow Pipeline code on Argo, including the API backend and UI all written with Argo dependency
- Internal IBM team divided to attack different areas: Compiler (Christian Kadner), API (Tommy Li), UI (Andrew), Feng Li (IBM Systems, China)
- Inability of Kubeflow Pipeline backend to take multiple CRDs, which is the default model Tekton follows. So everything needed to be bundled in one Pipeline Spec
- Type check, workflow utils, and parameter replacement are heavily tied with Argo API. In addition, the persistent agent is watching the resources using the Argo API type.
- MLOps Sig in CD Foundation leveraged to bring Kubeflow Pipelines and Tekton team together





OPENSIFT



kubernetes



## Pluggable Components

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

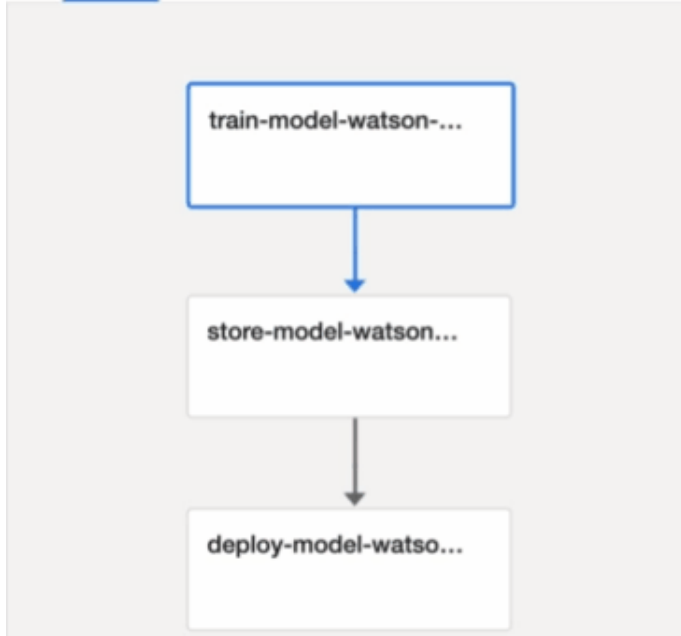


# Same KFP Experience: DAG, backed by Tekton YAML

Pipelines + Create run   + Upload version   + Create experiment   Delete

← default-watson-ml (default-watson-ml)

Graph   YAML



✕ train-model-watson-machine-learning

### Input parameters

compute_name	
compute_nodes	
execution_command	
framework	
framework_version	
run_definition	
run_name	
runtime	
runtime_version	
train_code	

### Output parameters

run-uid	/tmp/outputs/run_uid/data
training-uid	/tmp/outputs/training_uid/data

### Arguments

Show summary   Static pipeline graph



# Same KFP Exp: Logs, Lineage Tracking and Artifact Tracking

Experiments > tekton-experiments

Run of watson-ml-pipeline-with-artifacts (d6bd5)

Retry Clone run Terminate Archive

Graph Run output Config

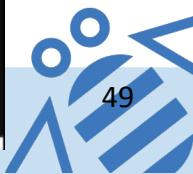
```
graph TD; A[create-secret-ku...] --> B[train-model-wats...]; B --> C[store-model-wats...]; C --> D[deploy-model-wa...];
```

Runtime execution graph. Only steps that are currently running are highlighted.

kfp-on-wml-training-run-1dd60-train-model-watson-machine--xt4gc

Input/Output Visualizations ML Metadata Volumes **Logs** Pod Events

```
9
10
11
12
13
14 -----
15 Log monitor done.
16 -----
17
18
19
20
21 #####
22
23 Metric monitor started for training run: af80b10e-12f3-4053-a71c-31ff4ea8df56
24 #####
25 #####
26
27
28
29
30 -----
31 Metric monitor done.
32 -----
33
34
35 status: {'state': 'pending'}
36 {'completed_at': '2020-07-06T21:15:15.208Z', 'message': {'text': 'Training job af80b10e-12f3-4053-a71c-31ff4ea
37 training_details {'metadata': {'created_at': '2020-07-06T21:11:38.049Z', 'guid': 'af80b10e-12f3-4053-a71c-31ff
38
```



# End to end Kubeflow Components : With KFP-Tekton

Recurring run configs  
0 active  
[Manage](#)

Experiment description

## Runs

+ Create run

+ Create recurring run

Compare runs

Clone run

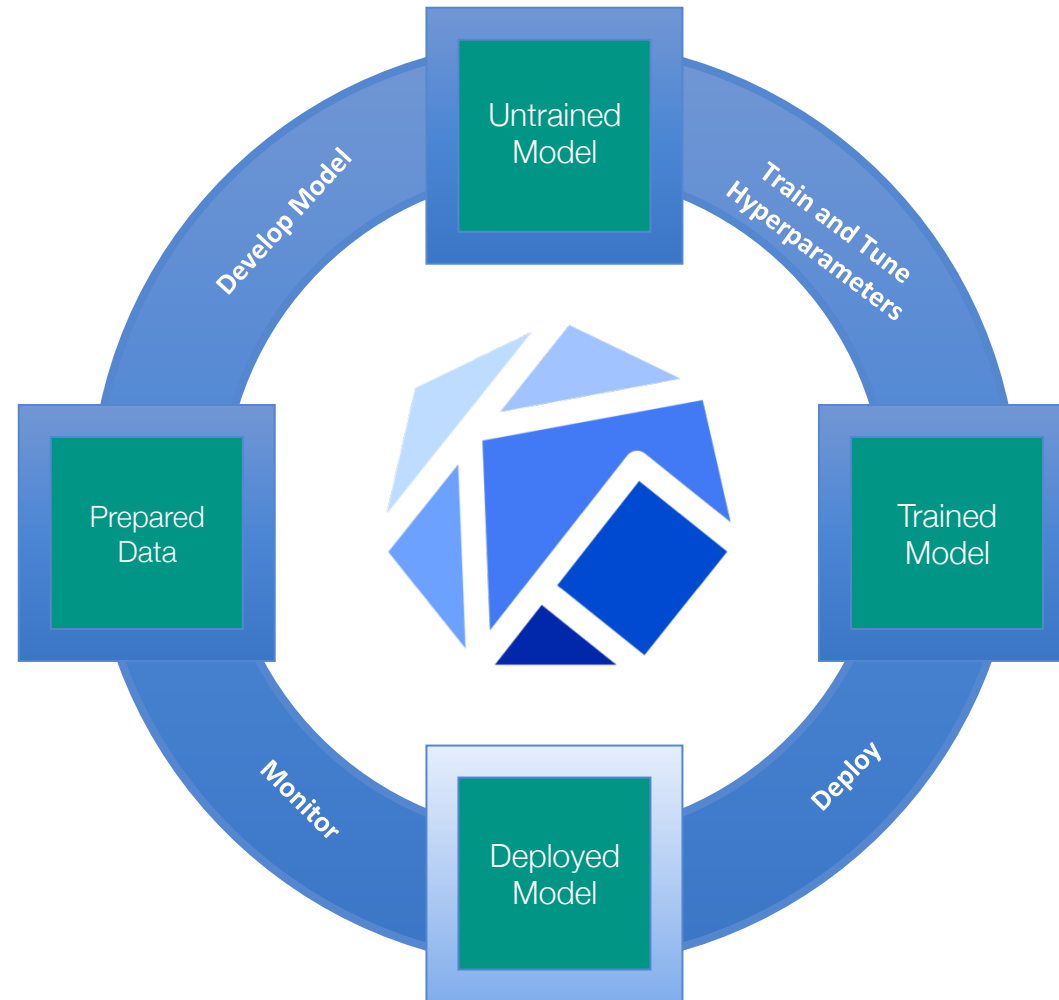
Archive

Filter runs



<input type="checkbox"/>	Run name	Status	Duration	Pipeline Version	Recurring Run...	Start time ↓
<input type="checkbox"/>	Run of mnist-e2e-pipeline (7d2c8)	✓	-	mnist-e2e-pipeline	-	7/7/2020, 12:28:38 AM
<input type="checkbox"/>	Run of mnist-model-cleanup (91455)	✓	-	mnist-model-cleanup	-	7/6/2020, 5:27:54 PM
<input type="checkbox"/>	mnist-e2e-pipeline-animesh (bf69b)	✓	-	mnist-e2e-pipeline	-	7/6/2020, 4:48:15 PM
<input type="checkbox"/>	Run of watson-ml-pipeline-with-artifacts (d...	✓	-	watson-ml-pipeline-with-arti...	-	7/6/2020, 2:11:07 PM
<input type="checkbox"/>	Run of watson-ml-pipeline-with-artifacts (d...	✓	-	watson-ml-pipeline-with-arti...	-	6/22/2020, 6:21:28 PM
<input type="checkbox"/>	Watson-ml-pipeline-with-artifacts	✓	-	watson-ml-pipeline-with-arti...	-	6/14/2020, 7:15:30 PM
<input type="checkbox"/>	▲ Run of watson-ml-pipeline (f5876)	✓	-	-	-	6/11/2020, 4:23:45 PM
<input type="checkbox"/>		✓	-	-	-	6/2/2020, 5:19:25 PM



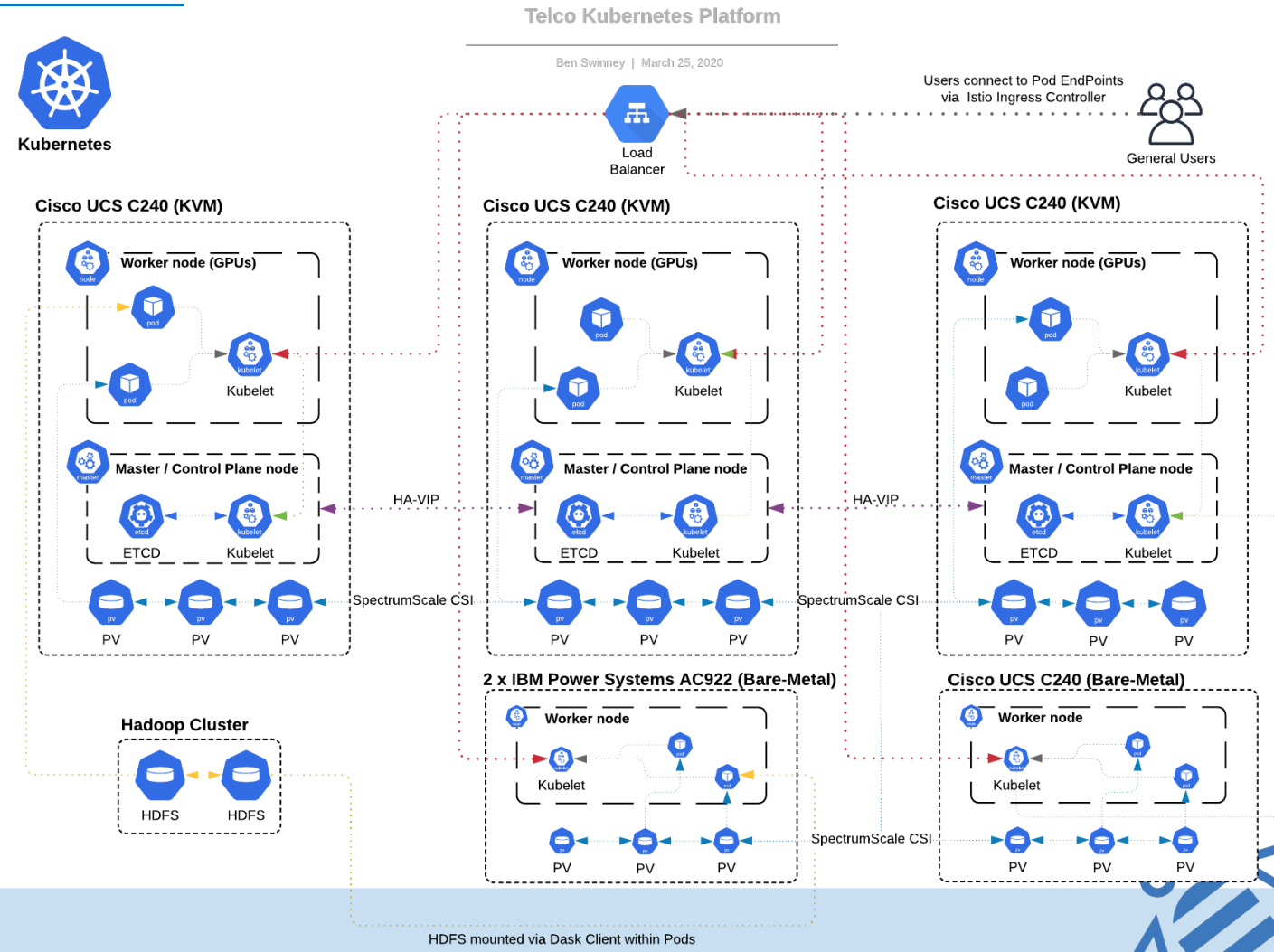


# Telstra: Collaborating with IBM to build an Open Source based OneAnalytics Platform leveraging Kubeflow

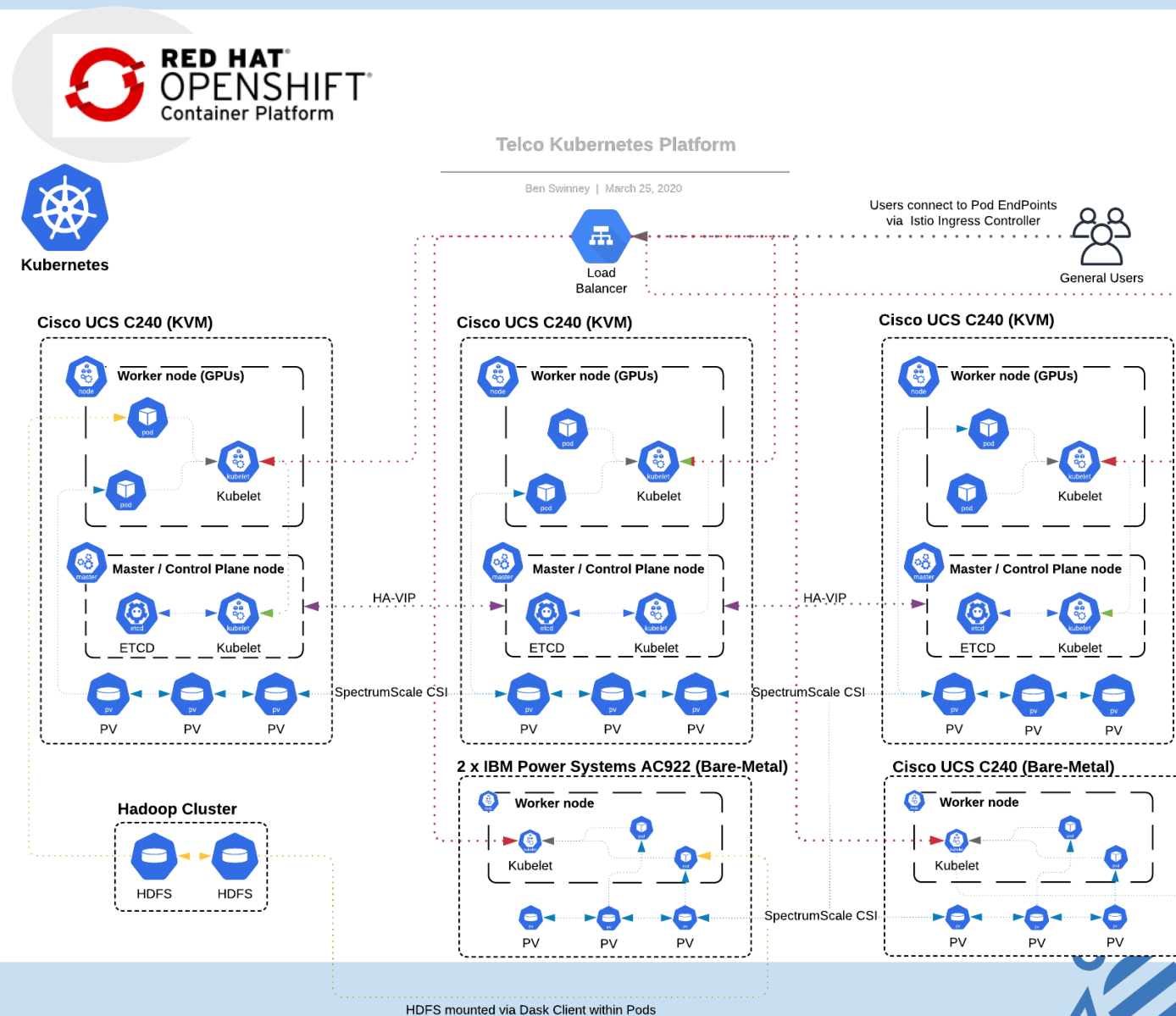
THINK 2020 Session: End-to-End Data Science and Machine Learning for Telcos: Telstra's Use Case  
<https://www.ibm.com/events/think/watch/replay/126561688>

## Telstra AI Lab - (TAIL) - Configuration

- Kubernetes – 1.15
- Spectrum Scale CSI Driver
- MetalLB for Load Balancing
- Istio 1.3.1 for ingress
- Kubeflow – 1.0.1
- Jupyter Notebook images are IBM’s multiarchitecture powerai images (<https://hub.docker.com/r/ibmcom/powerai/tags>)

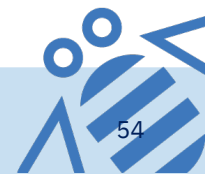
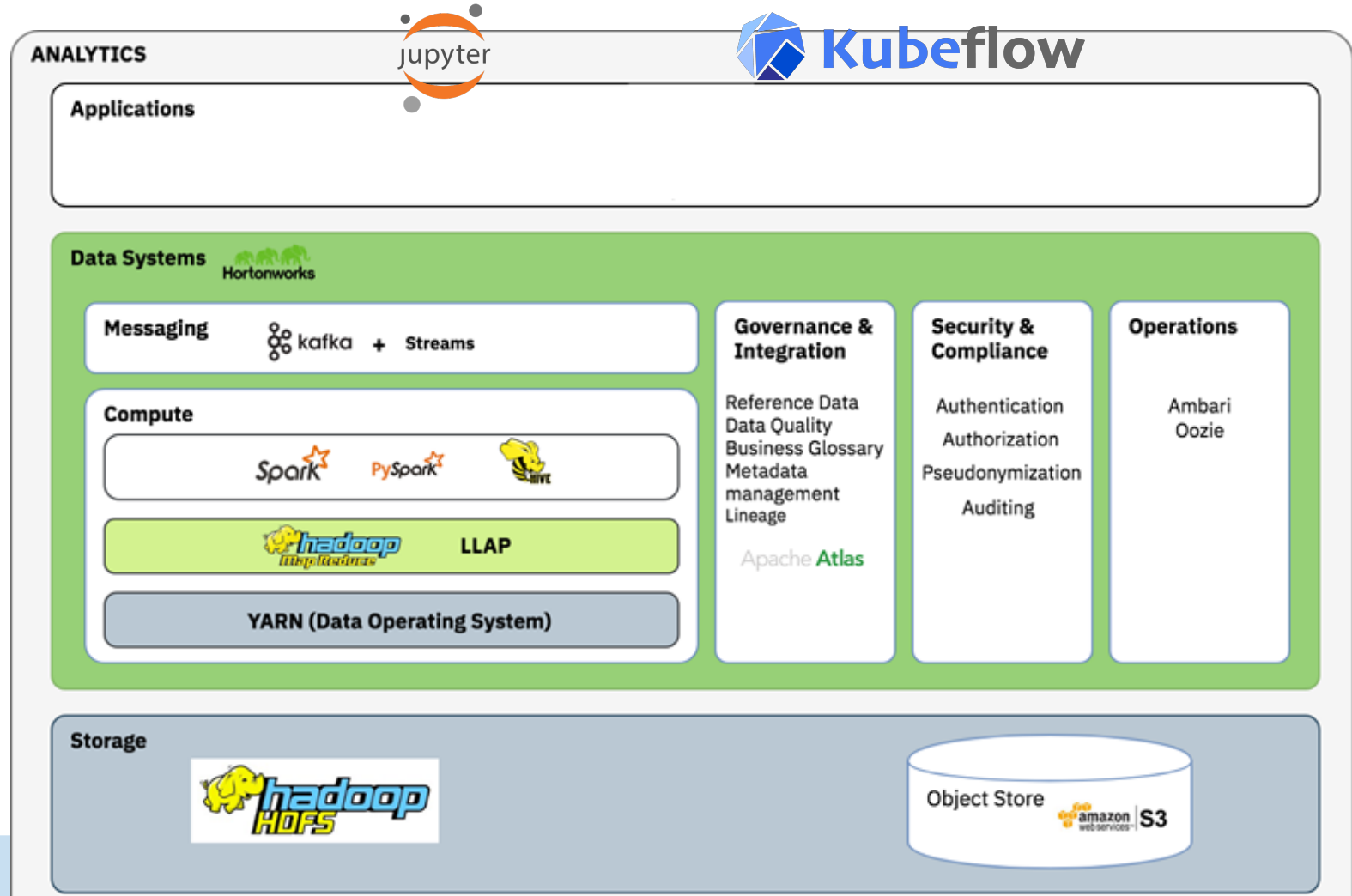


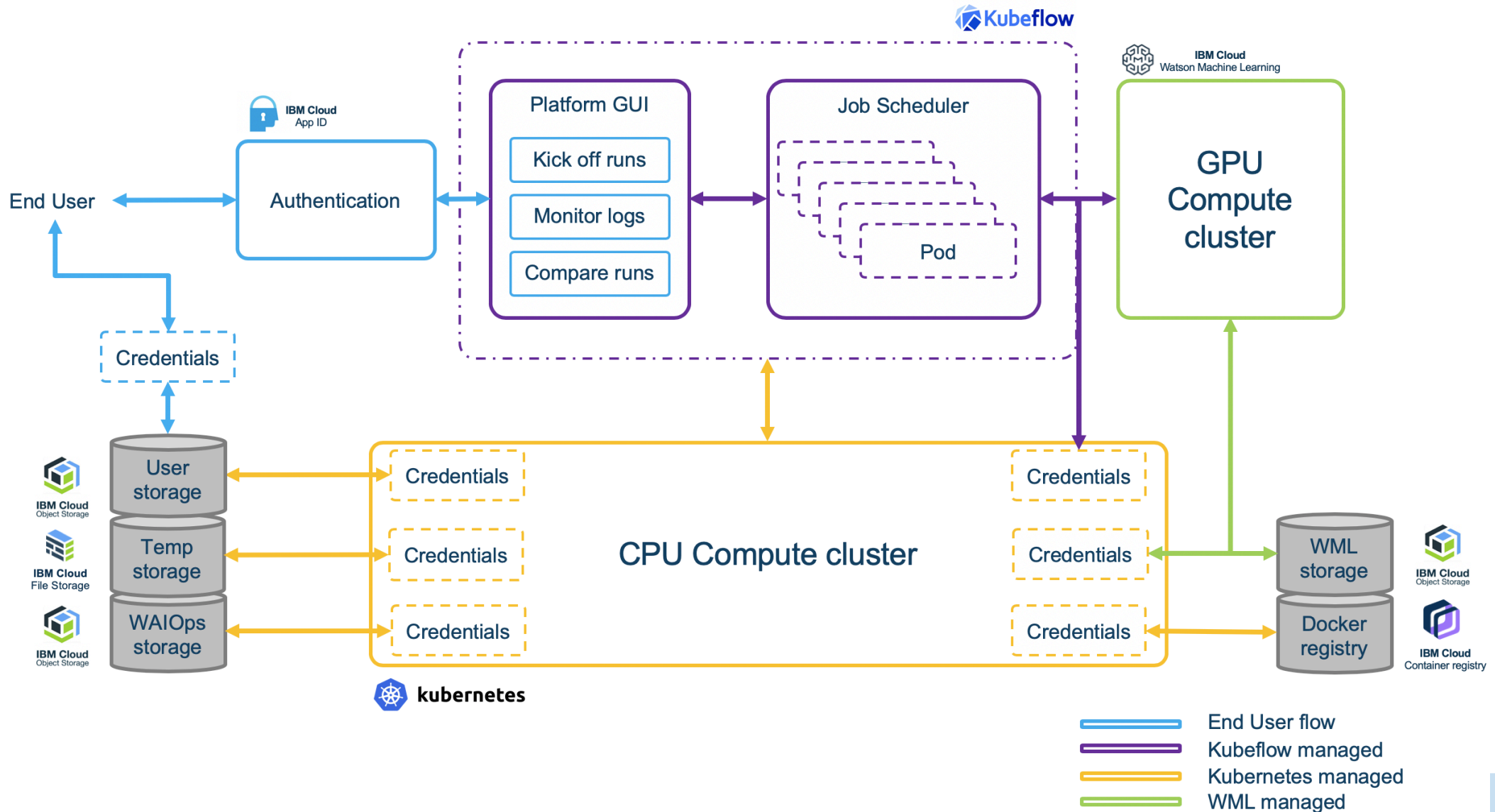
- RedHat Openshift – 4.3
- GPU Operator
- Kubeflow Operator
- Extending the compute
- Integrate feature stores and streaming technologies
- Integrate with CI/CD tools (Tekton Pipelines)





THINK 2020 Session: Enable Smart Farming using Kubeflow  
<https://www.ibm.com/events/think/watch/replay/126494864>





[https://stt-payload-kubeflow.us-east.containers.appdomain.cloud/\\_/pipeline-dashboard](https://stt-payload-kubeflow.us-east.containers.appdomain.cloud/_/pipeline-dashboard)

OpenAIHub

Pipelines

Experiments

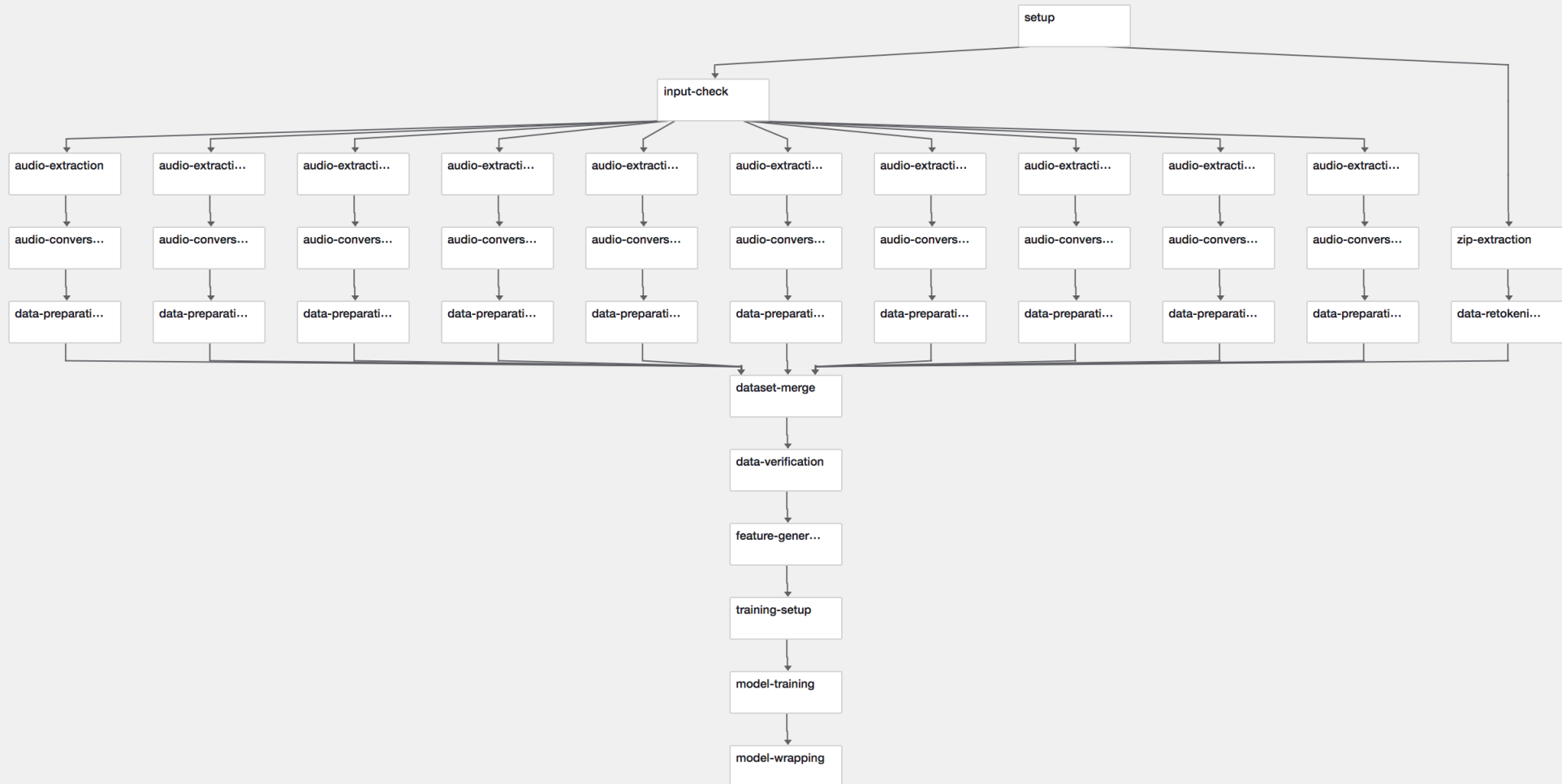
Notebooks

Archive

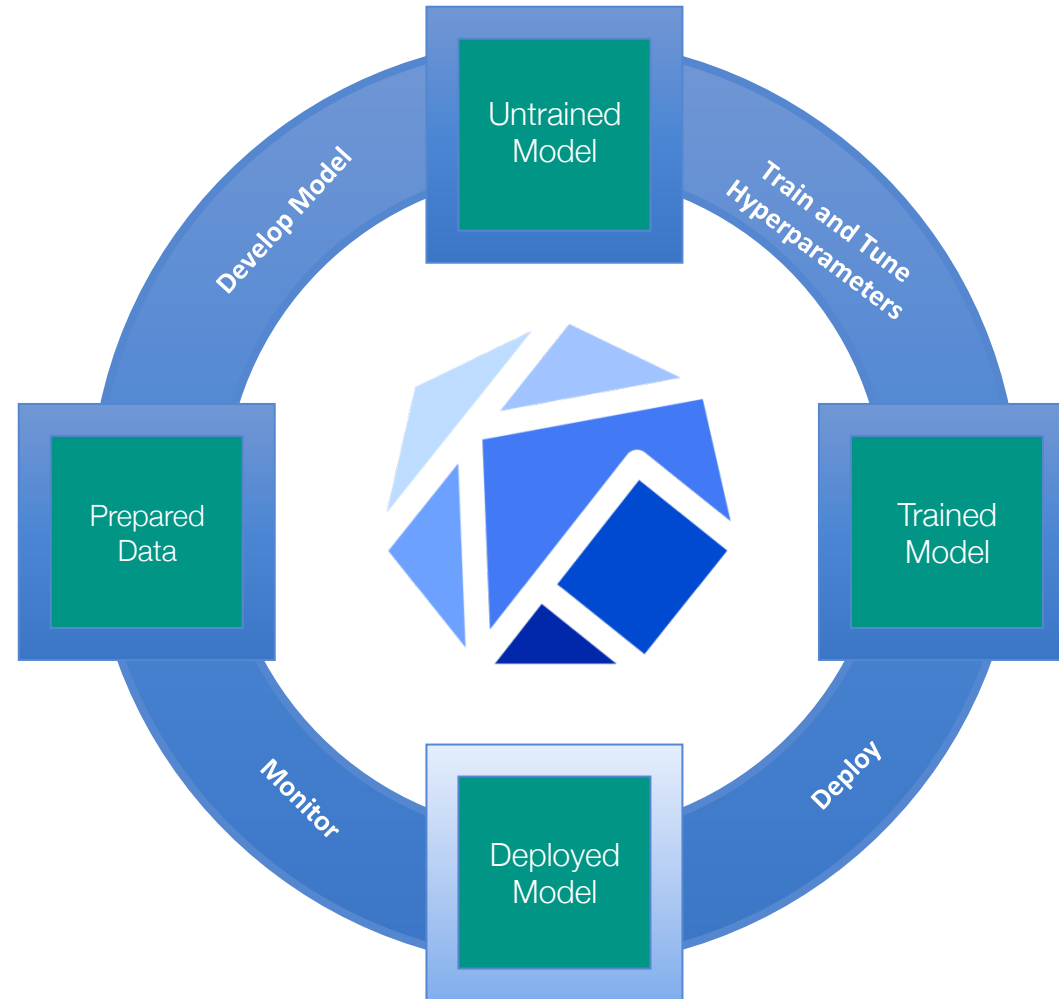
Watson Operations Pipeline

← Watson Operations Pipeline

Graph Source







'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.



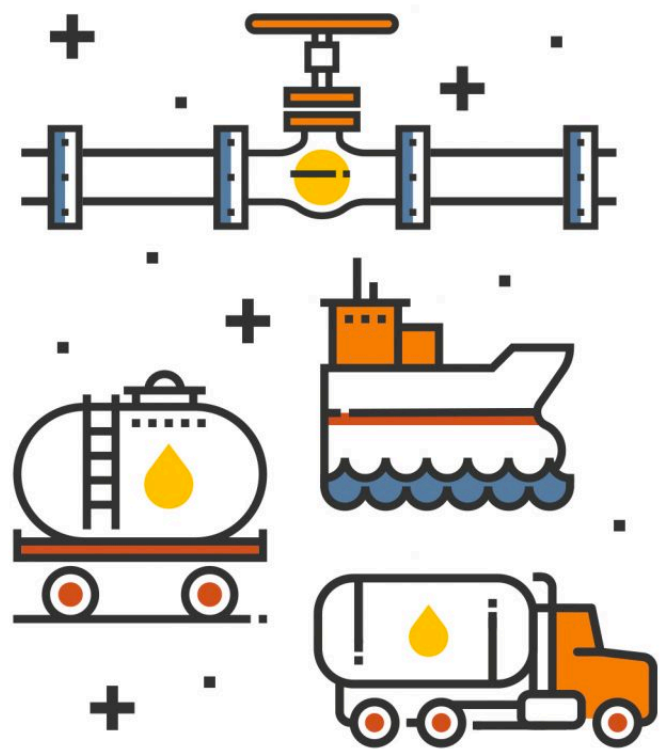
# Upstream, Midstream and Downstream

'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.

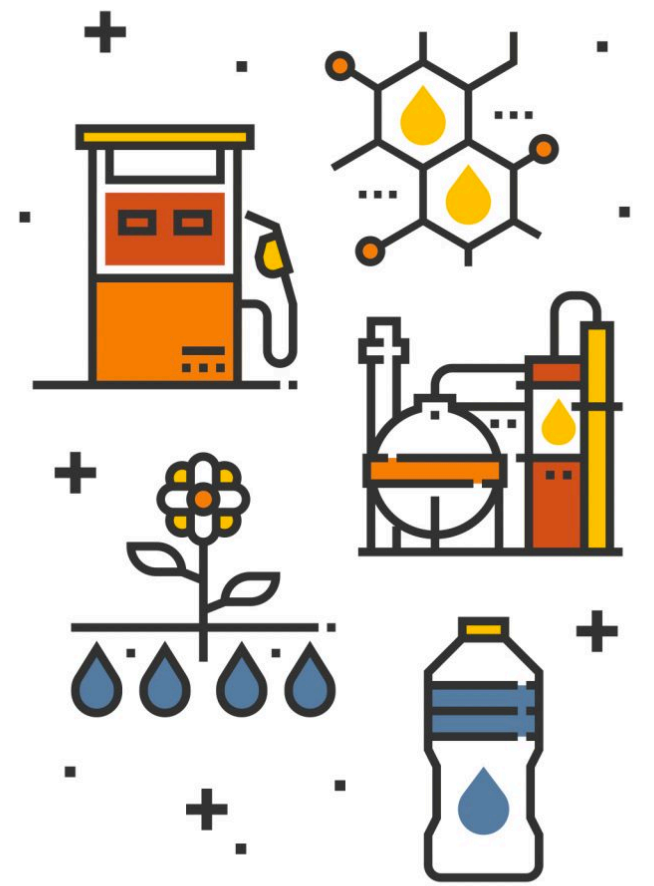
## UPSTREAM

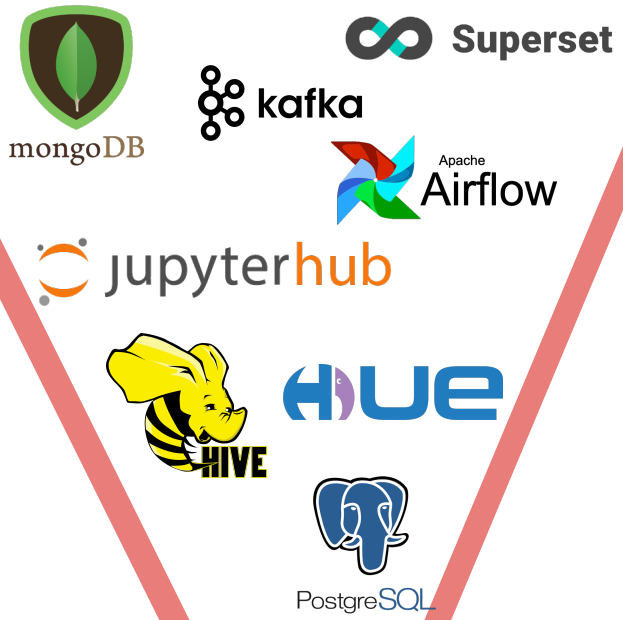


## MIDSTREAM



## DOWNSTREAM

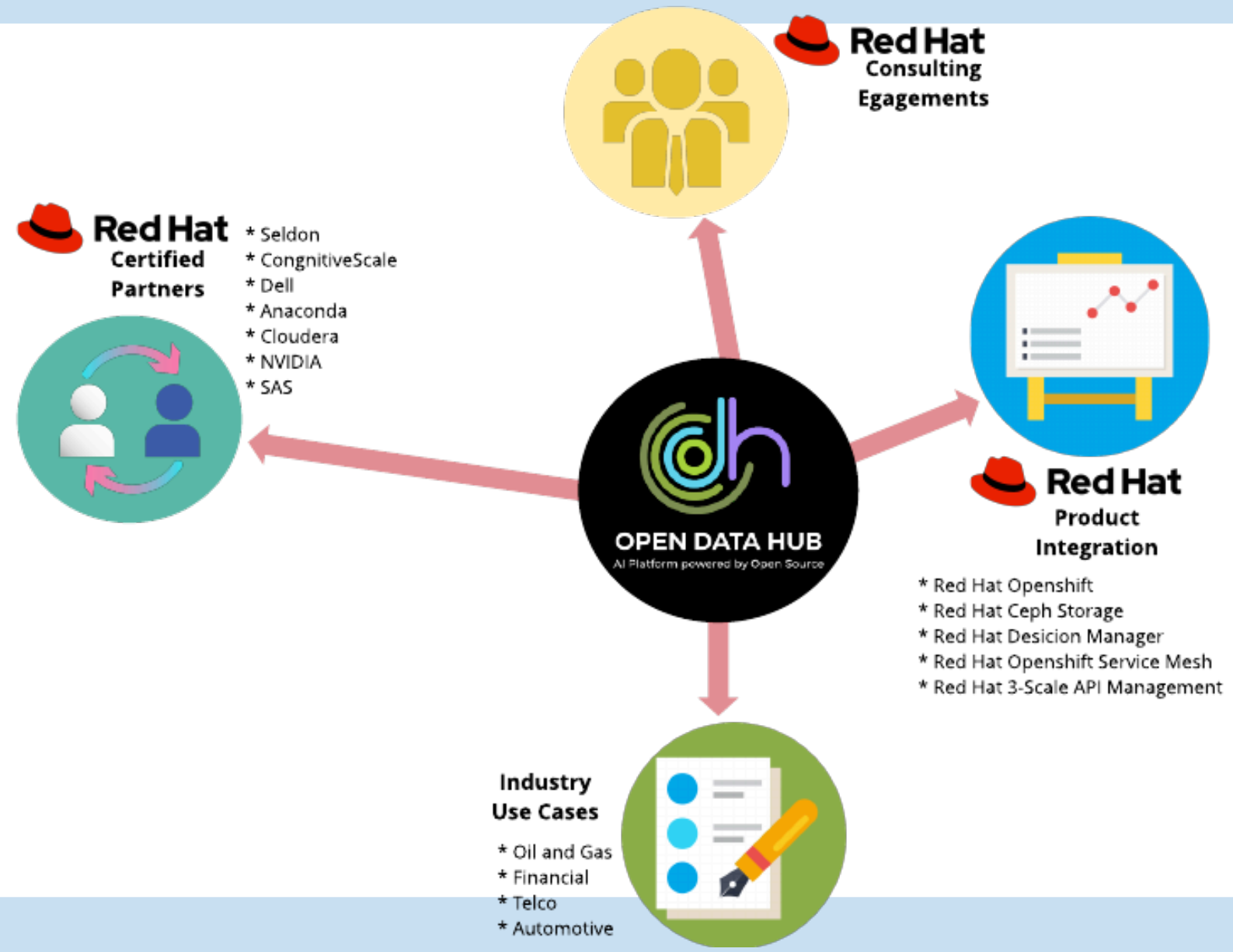




OpenShift  
Ready



### Data Platform





**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

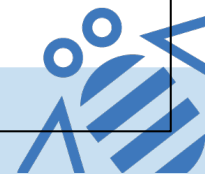
**Data in Motion**

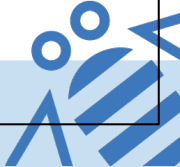
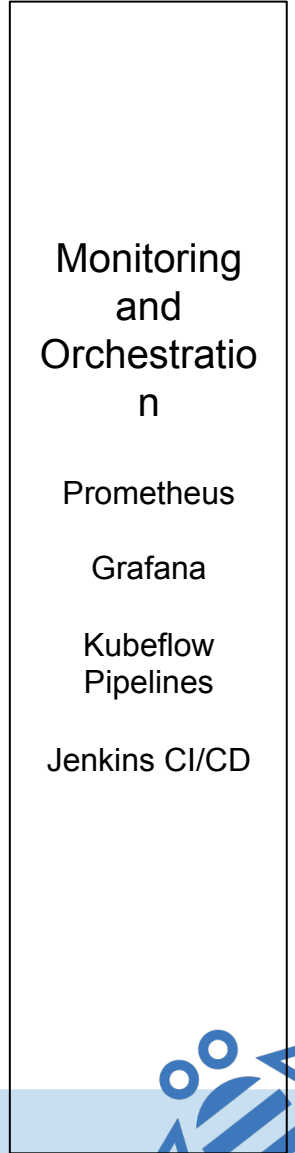
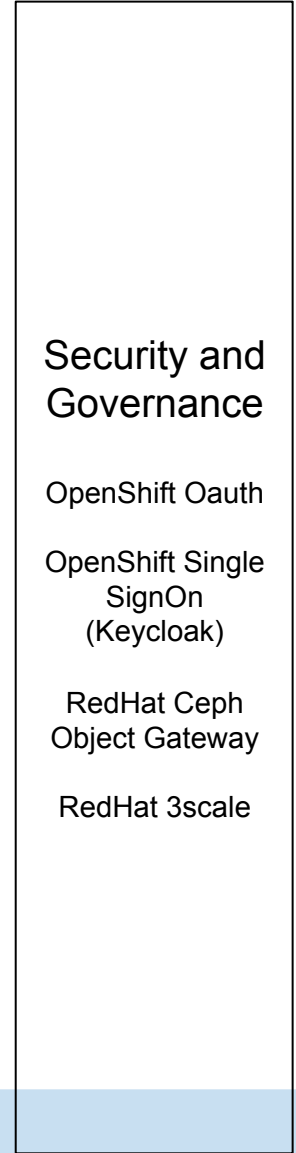
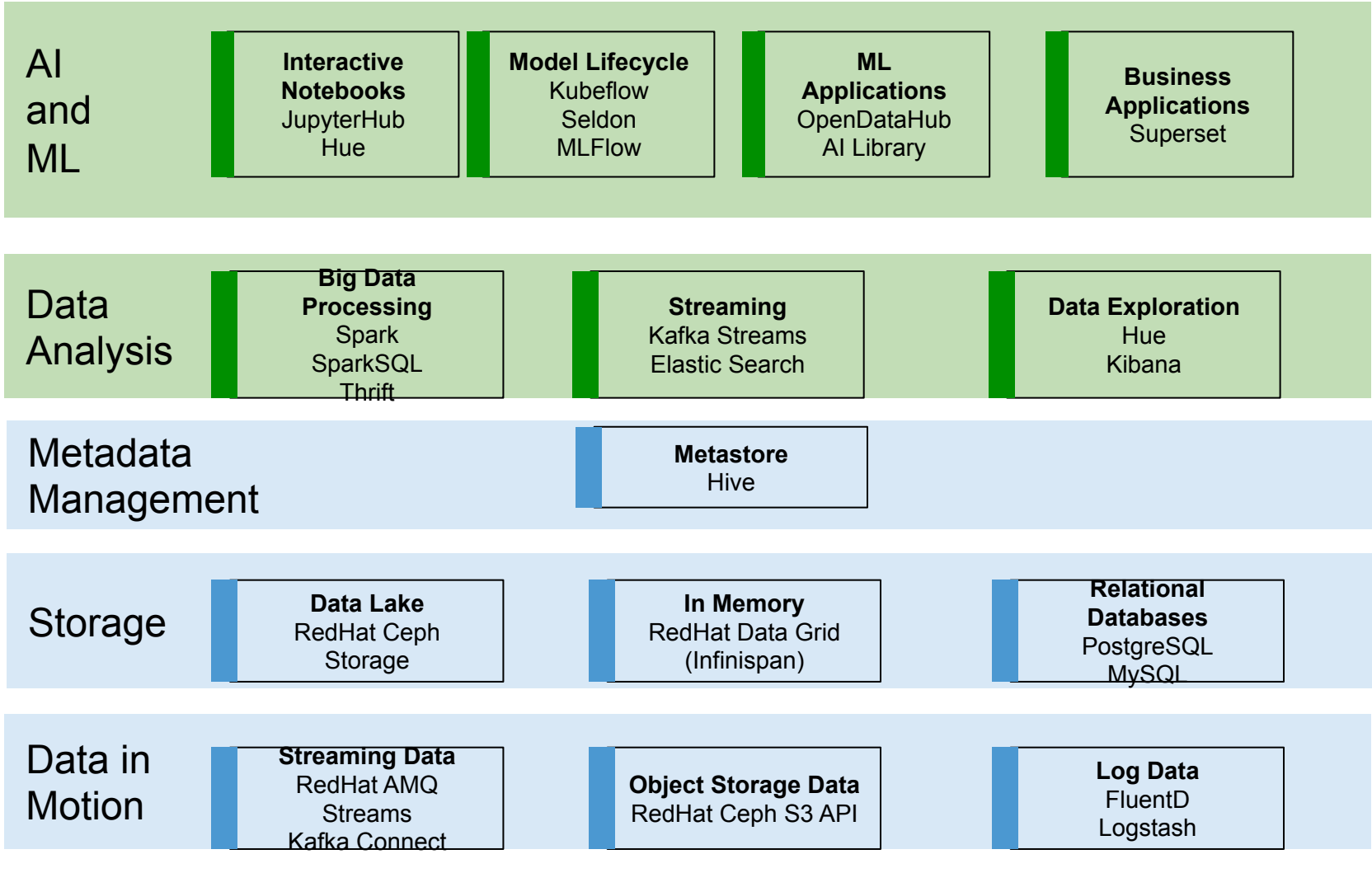
- Streaming Data
- Object Storage Data
- Log Data

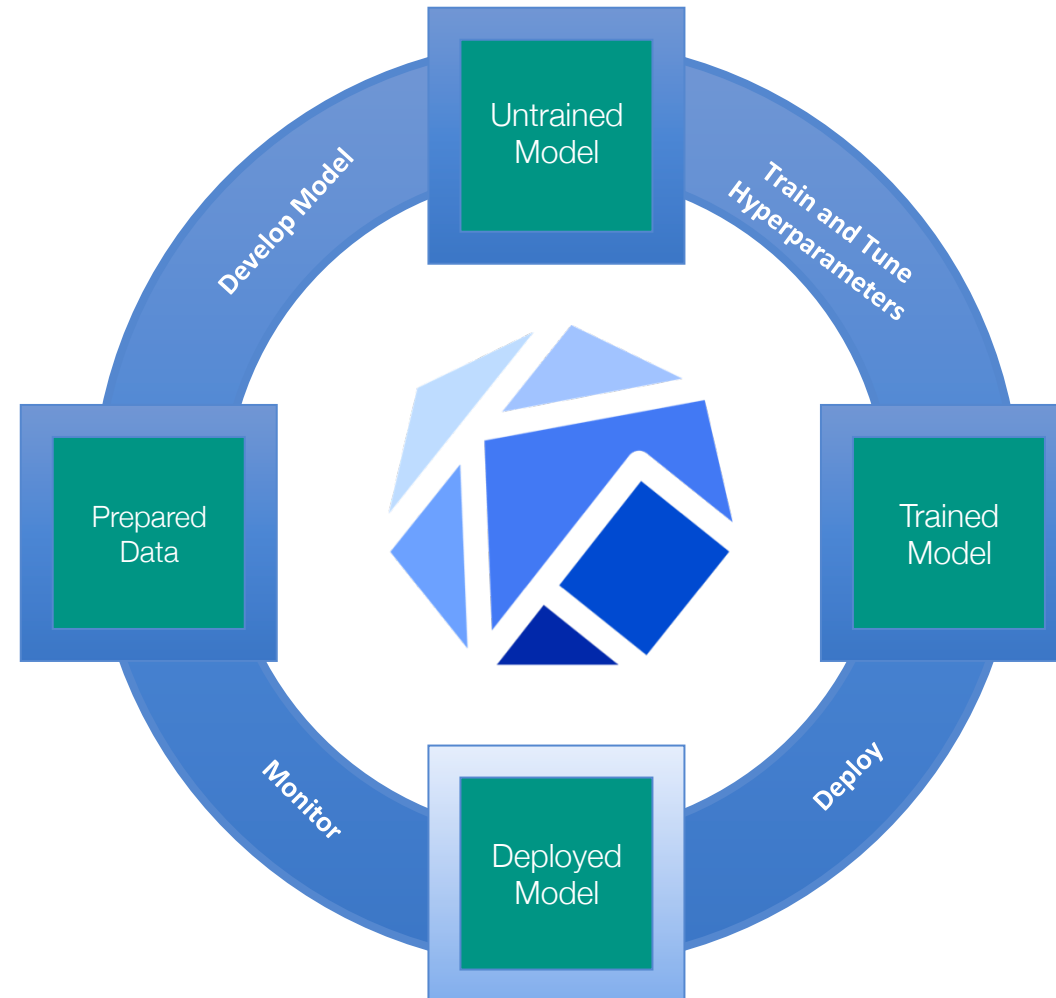
Security and Governance

Monitoring and Orchestration

**Red Hat**  
OpenShift Container Platform











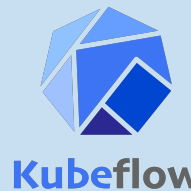
## Initial Goals:

- Kubeflow has a great traction, Make it available for OpenShift users  
Done in <https://github.com/opendatahub-io/manifests>
- Offer ODH users components installed by KF
- And offer components from ODH (Kafka, Apache SuperSet, Hive...) to KF community
- Decide if we can leverage KF project and community as upstream for ODH
- Think Kubernetes -> OpenShift
- Frees up ODH maintainers time to make sure KF keeps running well on OpenShift





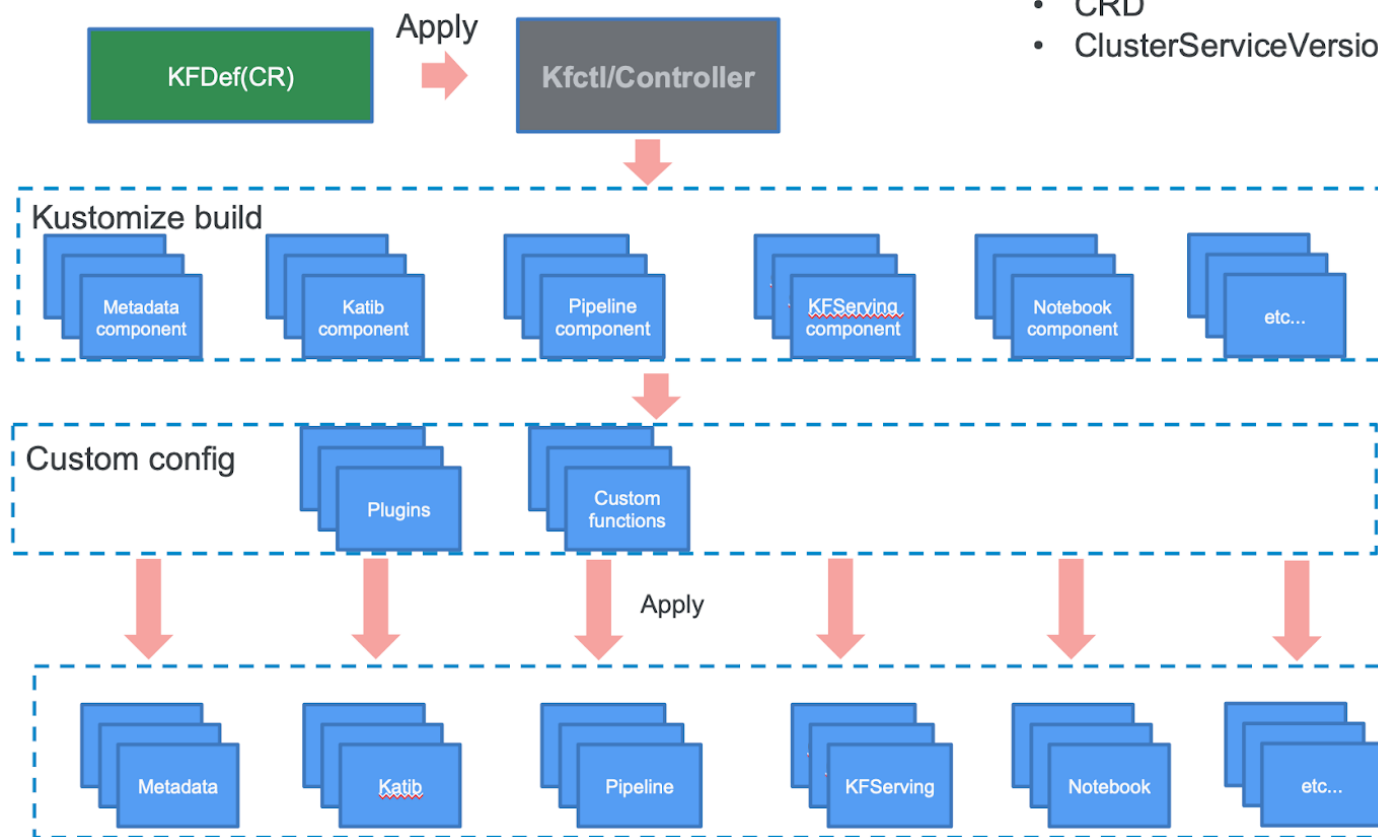
# Kubeflow Operator – Contributed by IBM to Kubeflow community to help enable OpenDataHub



- <https://operatorhub.io/operator/kubeflow>
- Deploy, manage and monitor Kubeflow
- On various environments
  - IBM Cloud
  - GCP
  - AWS
  - Azure
  - OpenShift
  - Other K8S

## KFCTL CONTROLLER - Initial deployment

- Controller deployment files
- CRD
  - ClusterServiceVersion



Kubeflow  
provided by IBM

Kubeflow Operator for  
deployment and management  
of Kubeflow



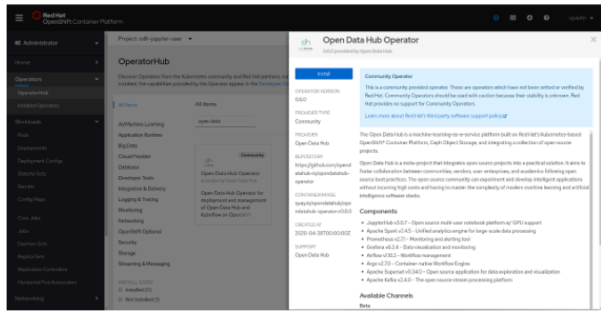
- A version of the Operator based on Kubeflow Architecture released: [https://developers.redhat.com/blog/2020/05/07/open-data-hub-0-6-brings-component-updates-and-kubeflow-architecture/?sc\\_cid=7013a000002DTqEAAW](https://developers.redhat.com/blog/2020/05/07/open-data-hub-0-6-brings-component-updates-and-kubeflow-architecture/?sc_cid=7013a000002DTqEAAW)
- Most of the components converted: <https://github.com/openshift/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



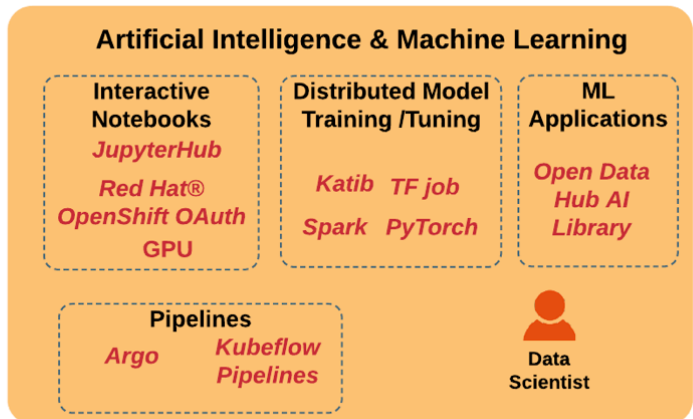
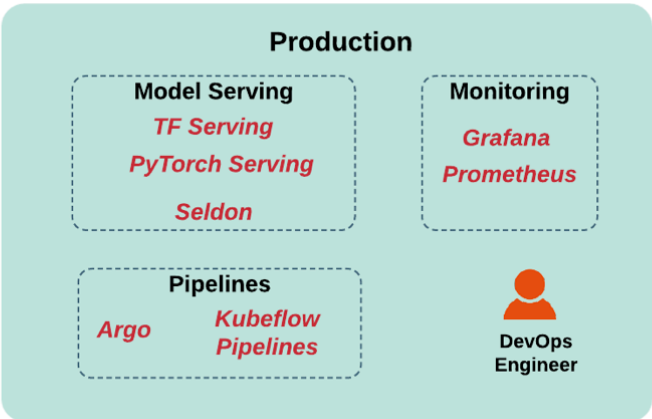
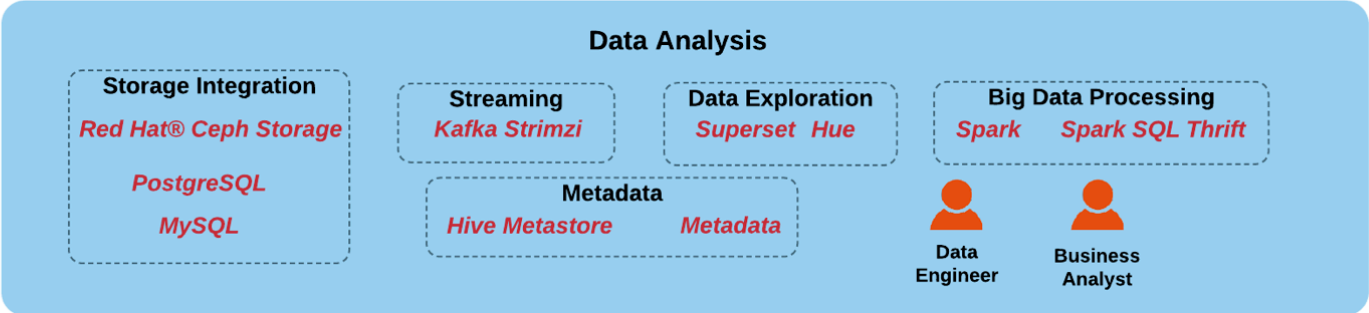
Open Data Hub (ODH) is a blueprint for building an AI-as-a-service platform on Red Hat's [Kubernetes](#)-based [OpenShift 4.x](#). Version 0.6 of Open Data Hub comes with significant changes to the overall architecture as well as component updates and additions. In this article, we explore these changes.





## OPEN DATA HUB

AI Platform powered by Open Source





# Open Data Hub in OpenShift

Home

Projects

Status

Search

Events

Catalog

Workloads

Networking

Storage

You are logged in as a temporary administrative user. Up

NAME ↑	STATUS
PR airflow-on-k8s-operator-system	Active
PR anonymous	Active
PR default	Active
PR kube-public	Active
PR kube-system	Active
PR opendatahub	Active

Home

Projects

Status

Search

Events

Catalog

Workloads

Networking

Storage

Builds

Monitoring

Compute

Administration

Project: opendatahub

jupyterhub-nb-kube-3aadmin

spark-operator

spark-operator, #1

strimzi

odh-message-bus-entity-operator, #1

odh-message-bus-kafka

odh-message-bus-zookeeper

superset

superset, #1

other resources

ailibrary-operator, #1

airflow-on-k8s-operator-controller-manager, #1

argo-server, #1

# IBM Apache Superset

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

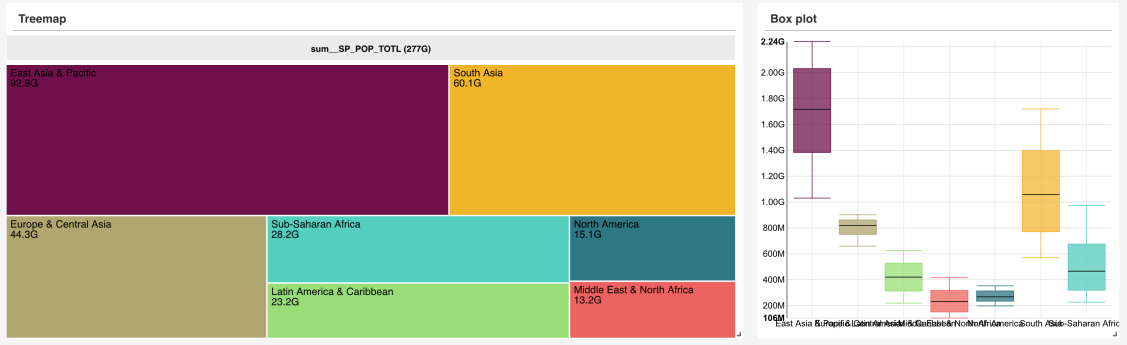
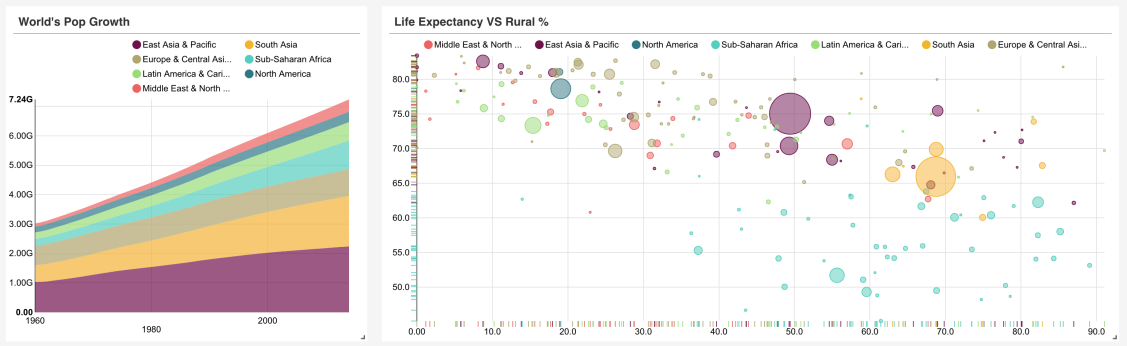
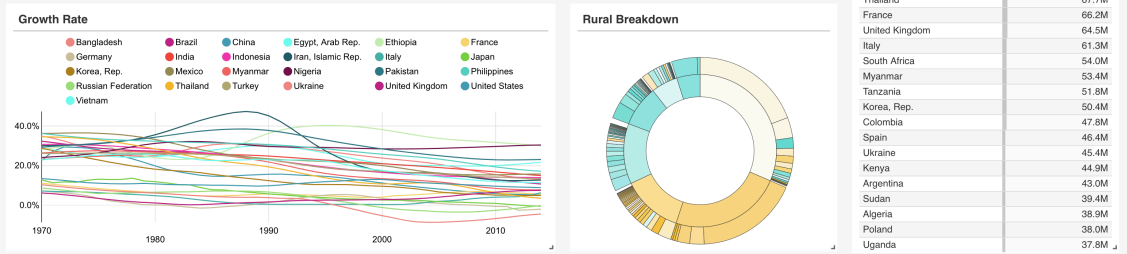
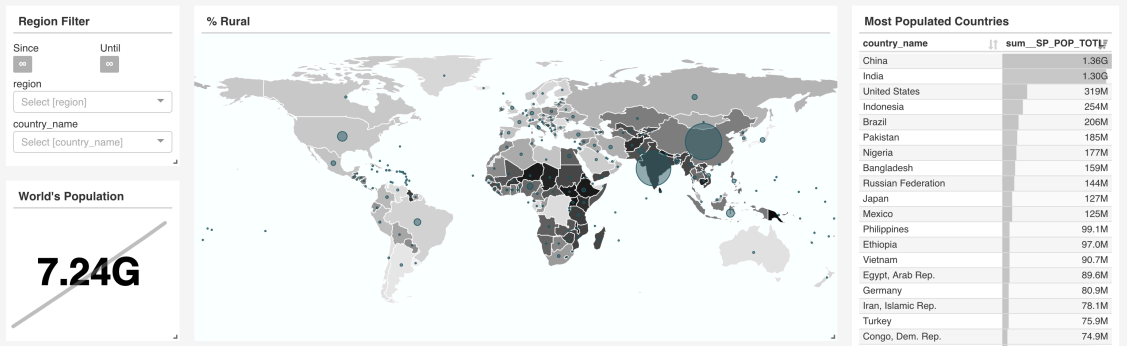
```
1 SELECT b.dashboard_id, a.dashboard_title, b.slice_id, c
2 FROM dashboards a
3 JOIN dashboard_slices b ON a.id = b.das
4 JOIN slices c on c.id = b.slice_id
```

Run Query Save Query Share Query

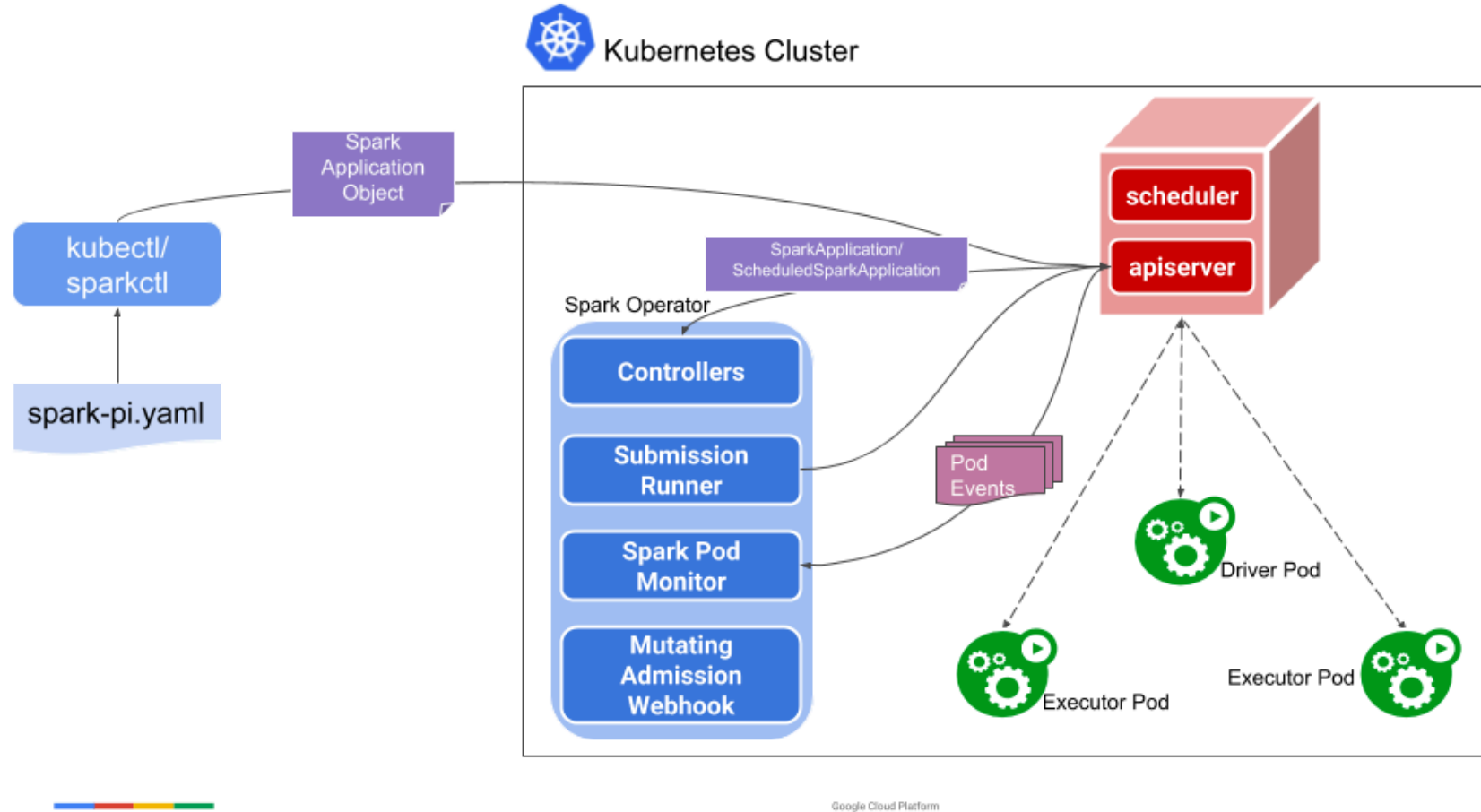
Visualize .CSV

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

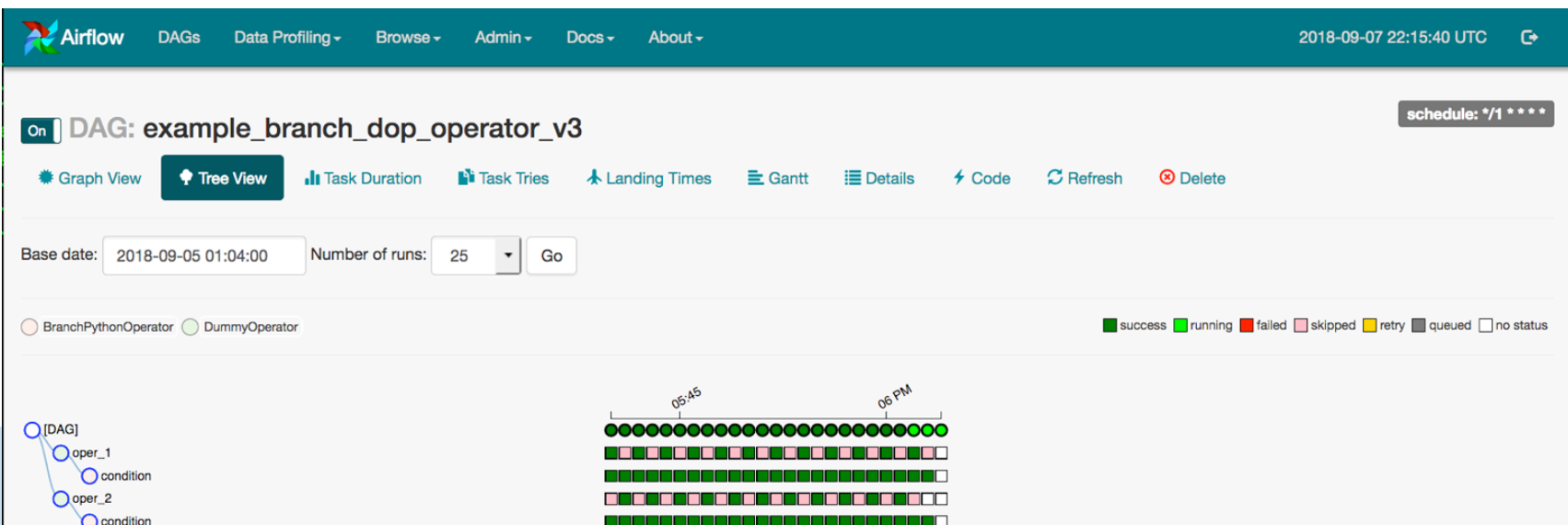
## World's Bank Data



- 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



- 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. )

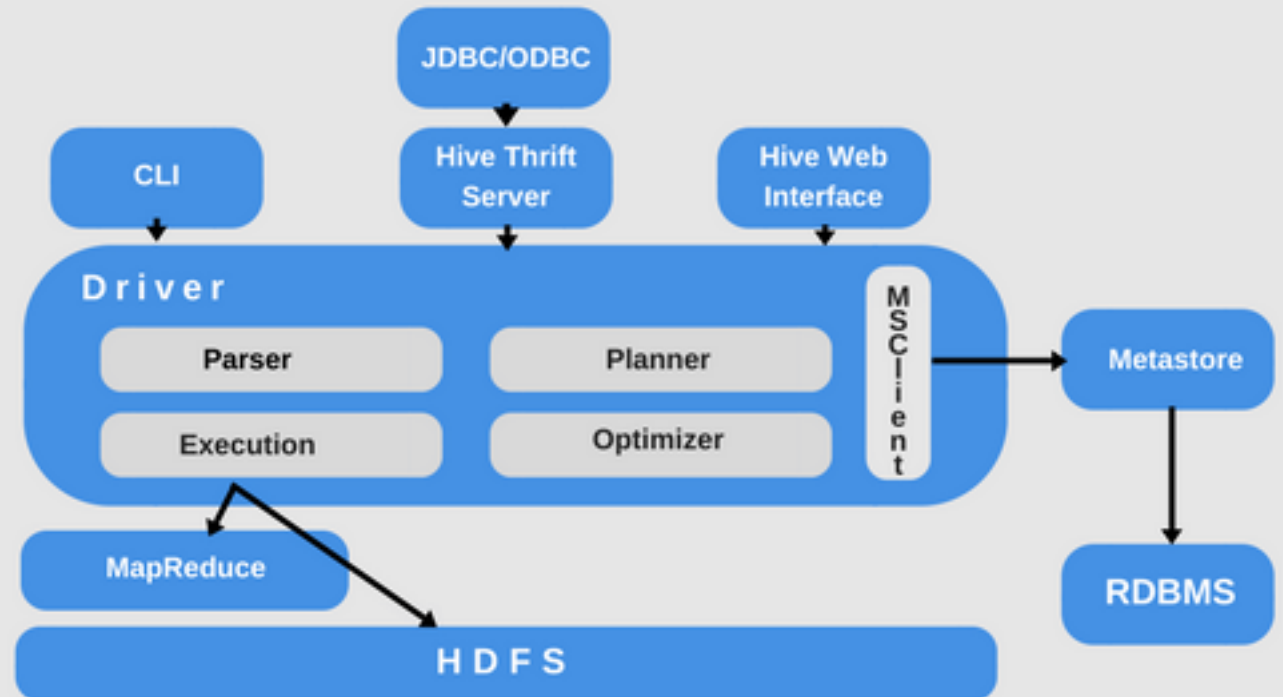


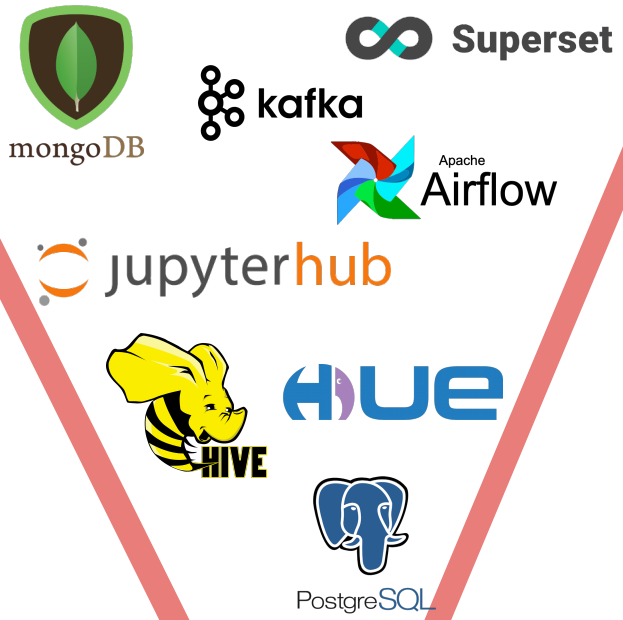




- Hive was one of the first abstraction engines to be built on top of MapReduce.
- Started at Facebook to enable data analysts to analyse data in Hadoop by using familiar SQL syntax without having to learn how to write MapReduce.
- Hive an essential tool in the Hadoop ecosystem that provides an SQL dialect 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.

## Apache hive Architecture

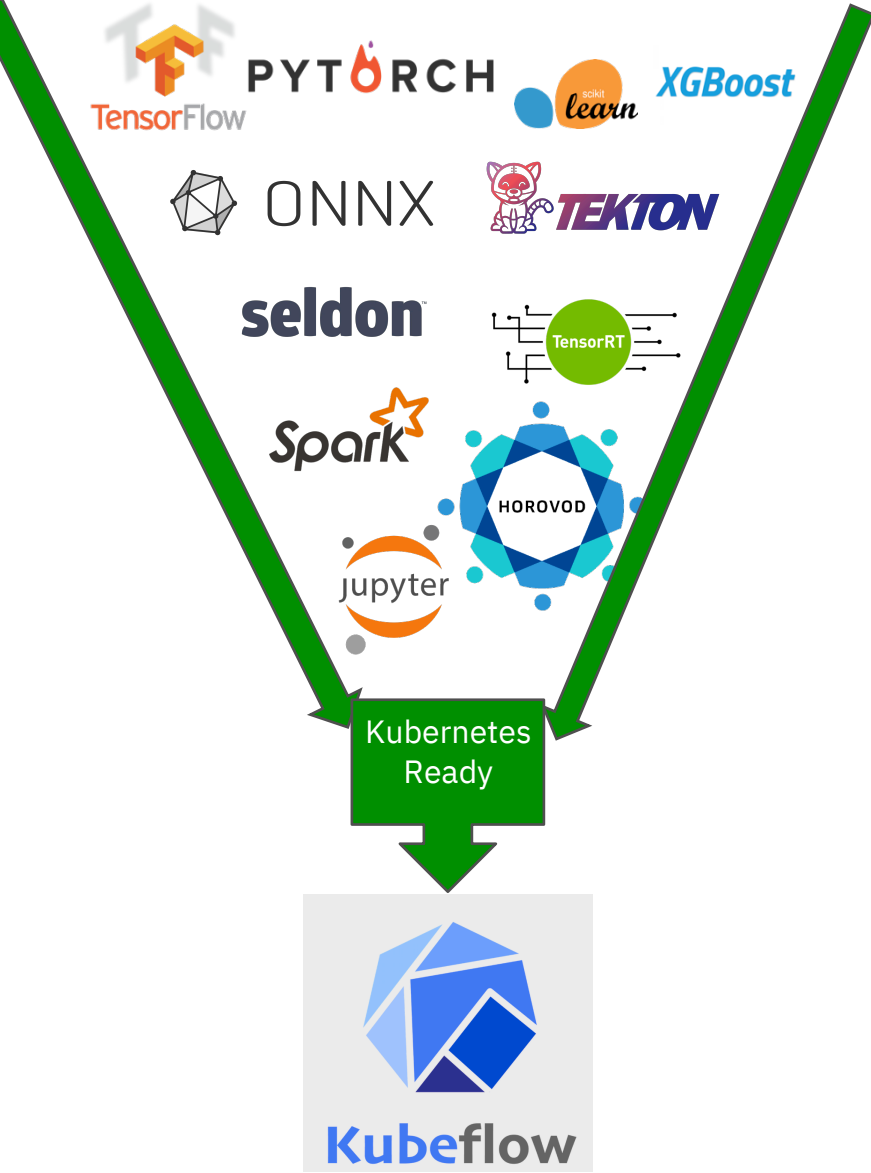




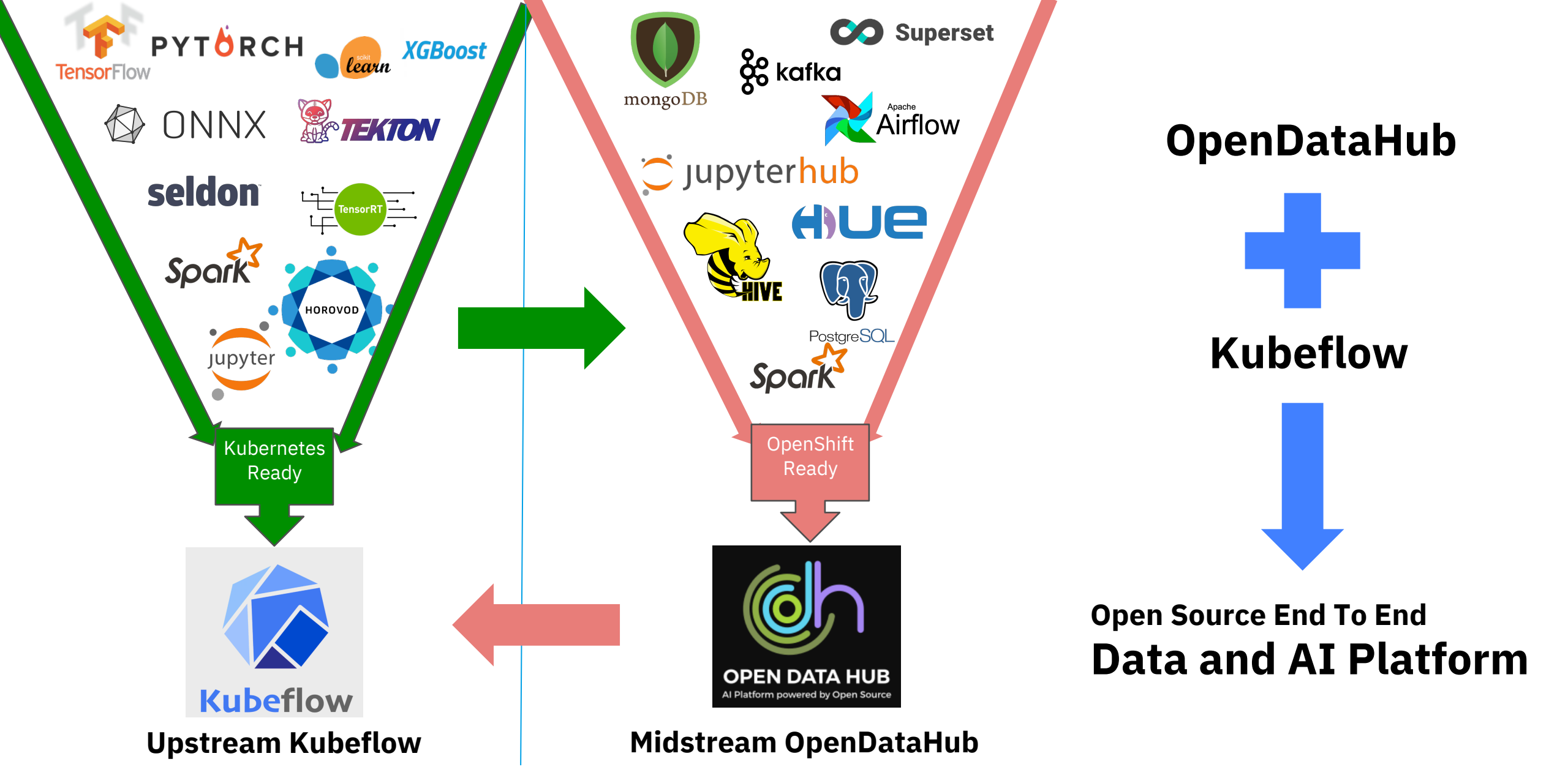
OpenShift  
Ready



### Data Platform



**ML and AI Platform**



Operator Hub - operatorhub.io

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

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

- Knowledge of Kubernetes, watch the dojo for Kubernetes project with the [IBM internal link](#) or [external link](#)
- Access to a Kubernetes cluster, either minikube or remote hosted
- Source code control and development with git and github, watch the presentation with the [IBM internal link](#) or [external link for git](#) and [external link for pull requests](#)
- Get familiar with go language, watch the introduction dojo with the [IBM internal link](#) or [external link](#)
- (optional) Knowledge of Istio and knative
- If you have more time,
  - Read [Kubeflow document](#) to learn more about Kubeflow project
  - Browse through Kubeflow [community](#) github



- Access to a Kubernetes cluster
  - minimal spec: 8vcpu, 16gb ram and at least 50gb disk for docker registry
- On IBM Kubernetes Service, provision the cluster with machine type b2c.4x16 and 2 worker nodes
- Follow Kubeflow [document](#) to have your cluster prepared
- On IKS cluster, follow this [link](#) to install the IBM Cloud CLI and helm followed by setting up IBM Cloud Block Storage as the default storage class



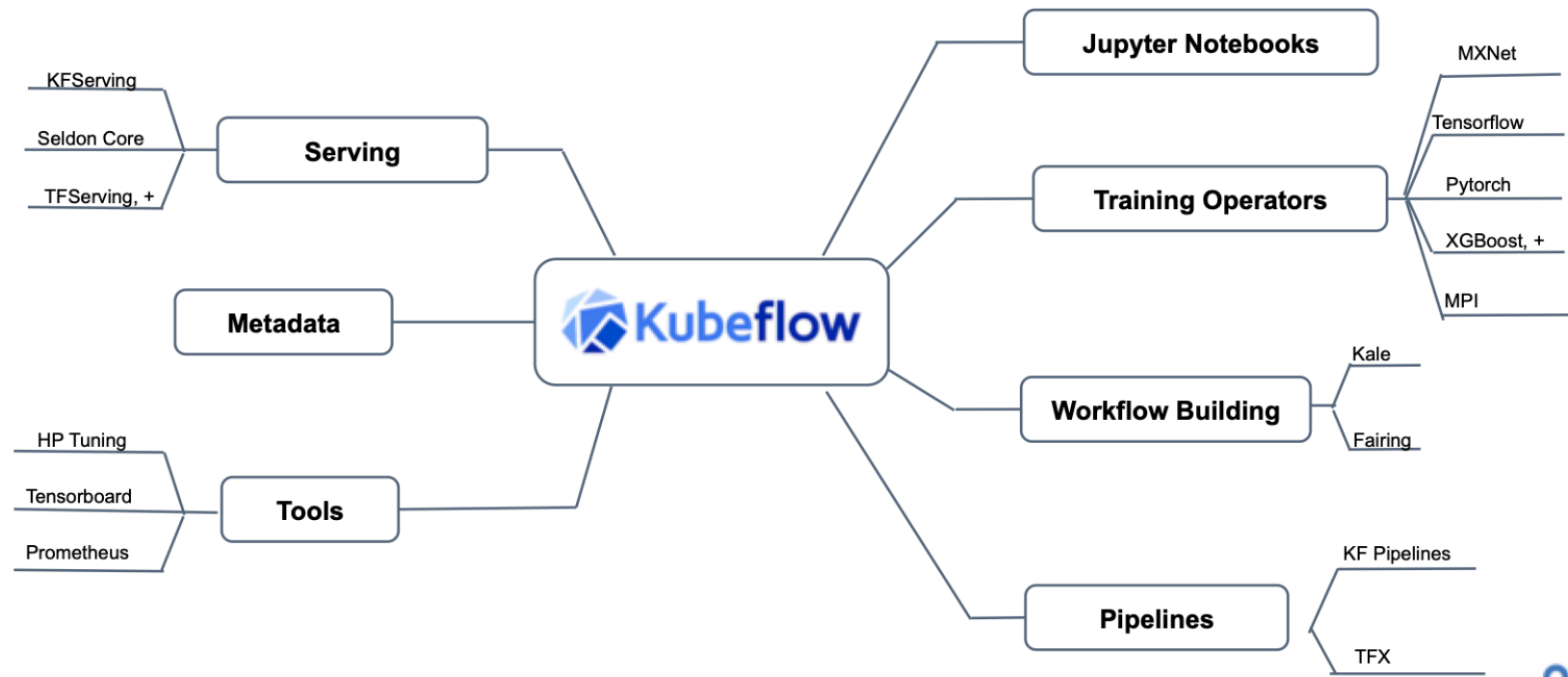
# Reach Out!

**Animesh Singh**

[singhan@us.ibm.com](mailto:singhan@us.ibm.com)

[twitter.com/AnimeshSingh](https://twitter.com/AnimeshSingh)

[github.com/AnimeshSingh](https://github.com/AnimeshSingh)



**Kubeflow Dojo: Live**  
**Dates: 15<sup>th</sup> and 16<sup>th</sup> July**

<https://ec.yourlearning.ibm.com/w3/event/10082348>

**Kubeflow Dojo: Virtual**  
[github.com/ibm/KubeflowDojo](https://github.com/ibm/KubeflowDojo)

