

# Optimizing resilience and availability by migrating from JupyterHub to the Kubeflow Notebook Operator

David Hoover

Alexander Perlman

## Who we are

### David Hoover

*Senior Lead Devops Engineer*

Capital One

<https://www.linkedin.com/in/david-hoover-705906169/>

### Alexander Perlman

*Senior Lead Software Engineer*

Capital One

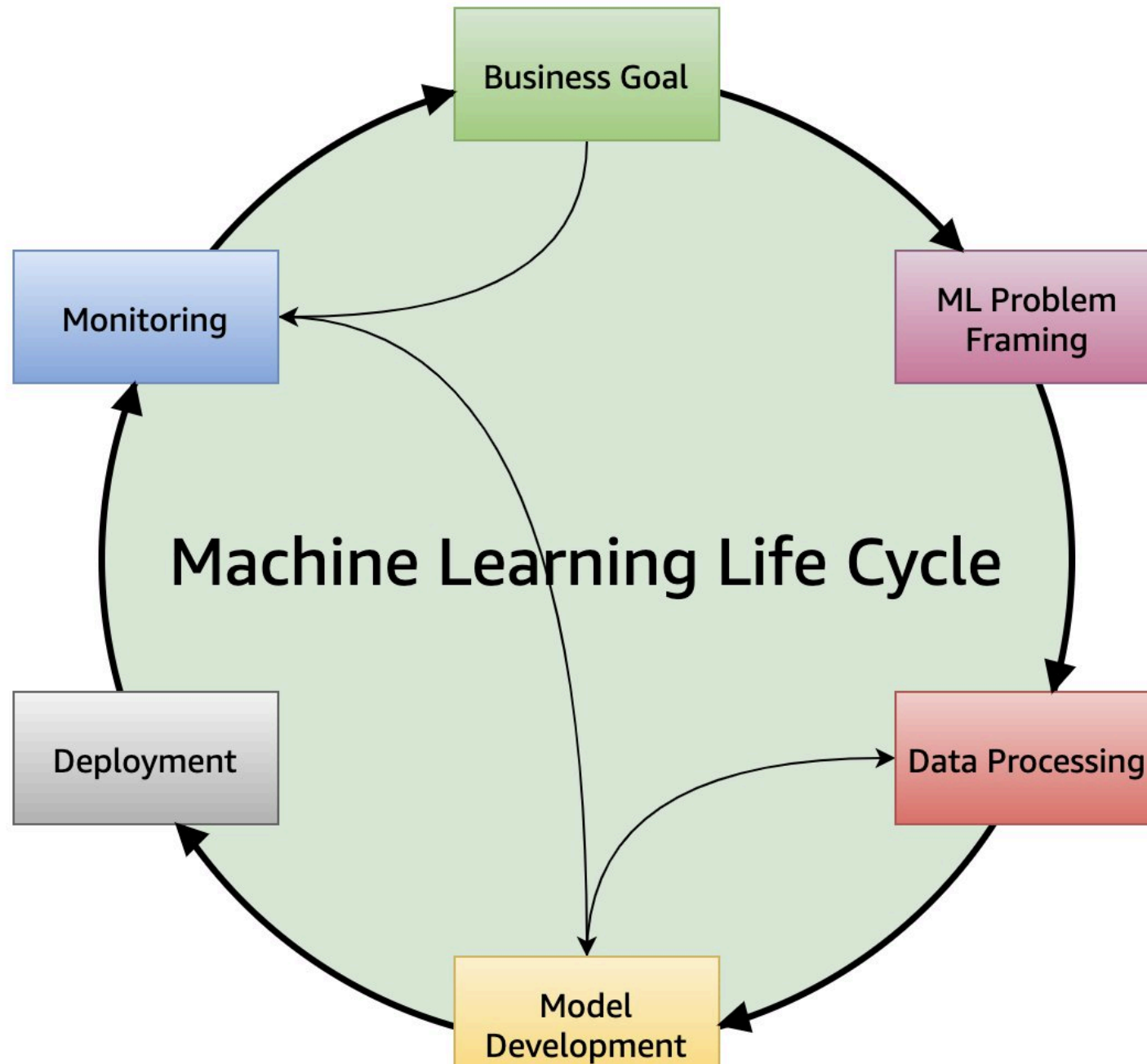
<https://www.linkedin.com/in/alexander-perlman-a396a654/>

<https://github.com/droctoThorpe>



The Jupyter logo consists of two thick orange curved lines forming a partial circle around the text. Four dark gray circles are positioned at the corners: top-left, top-right, and bottom-left are smaller, while the bottom-right one is larger.

jupyter



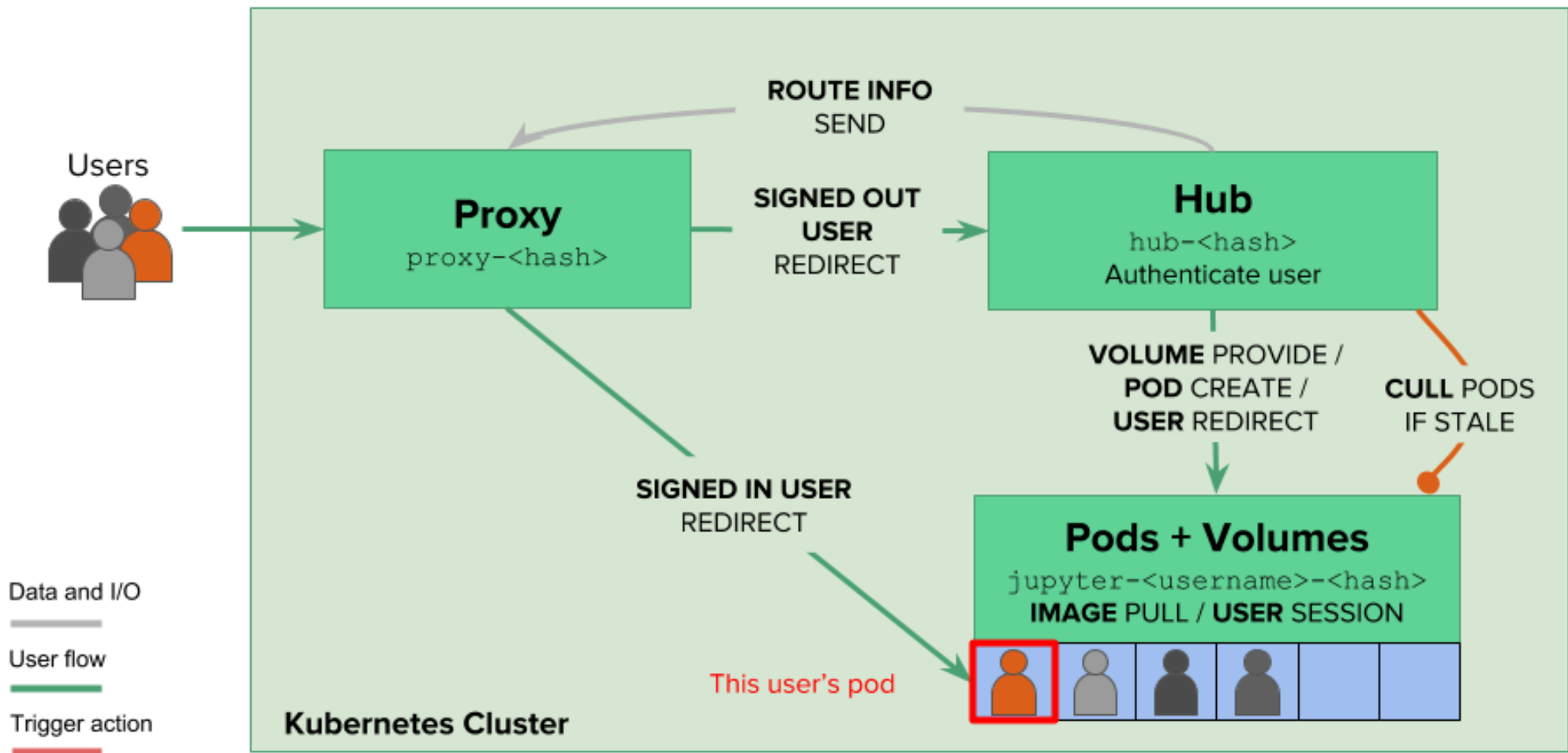


jupyterhub

# JupyterHub Architecture (high-level details)

**Cloud Volumes**  
Provides persistent storage

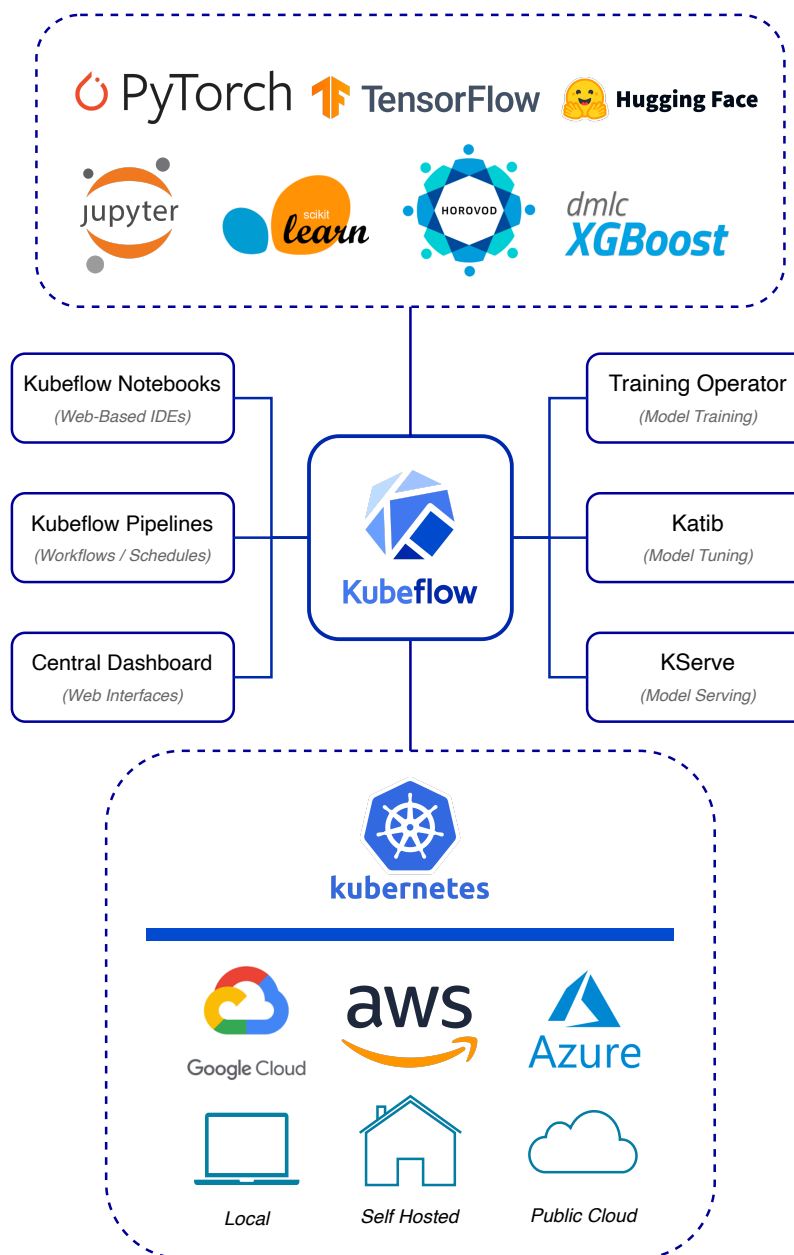
**Image Registry**  
Provides environment images





**Kubeflow**





## Operators

```
<pre class="mermaid"> %%{init: {'theme':'dark'}}%% graph LR; User(User)--creates-->CR(Custom Resource); subgraph Desired State Controller(Controller)--watches-->CR; CR--triggers-->Controller; end Controller--reconciles-->State(State); subgraph Actual State State; end </pre>
```

## Notebook Operator

```
<pre class="mermaid"> %%{init: {'theme':'dark'}}%% graph LR; User(User through UI)--creates-->Notebook(Notebook CR); Controller(Controller)--watches-->Notebook; Notebook--triggers-->Controller; Controller-->K8s(Kubernetes API); K8s-->StatefulSet; K8s-->Service; K8s-->VirtualService; </pre>
```

## Some context

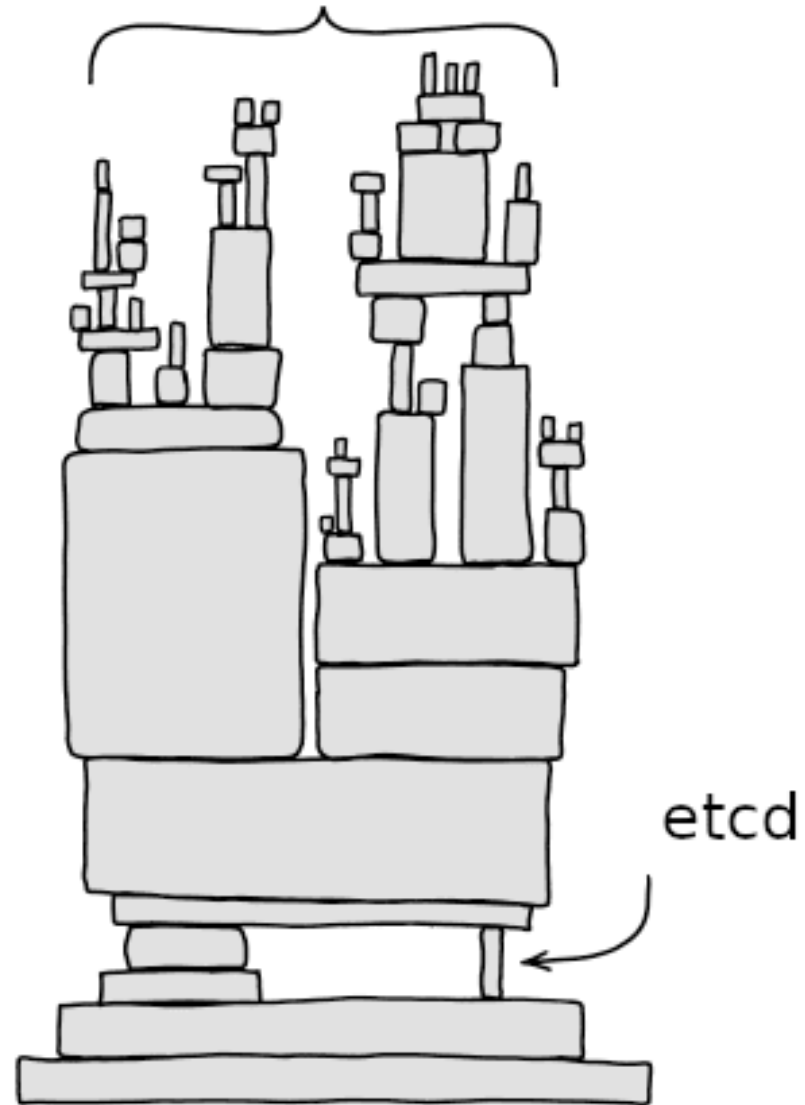
<https://github.com/kubeflow/kubeflow/issues/1630>

# Singleton

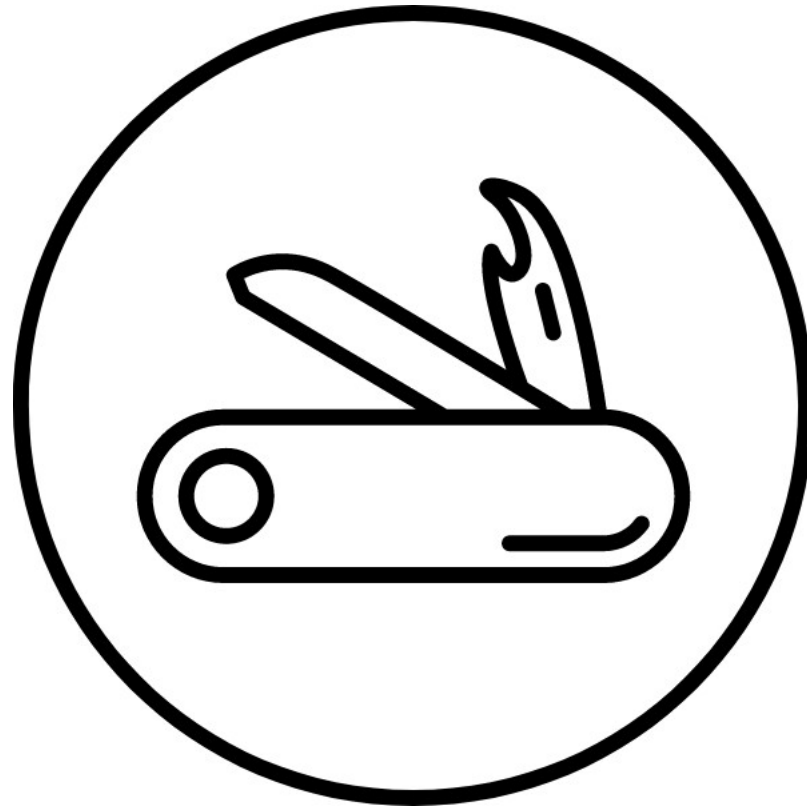
To quote one of the maintainers:

The Hub itself is a single-point-of-failure and must own the database it talks to. Other Hub instances may not be running and talking to the same database, or there will be errors and inconsistent state. So you cannot easily have two Hub replicas and failover between them.

# All Cloud Native Infrastructure



**Auth**

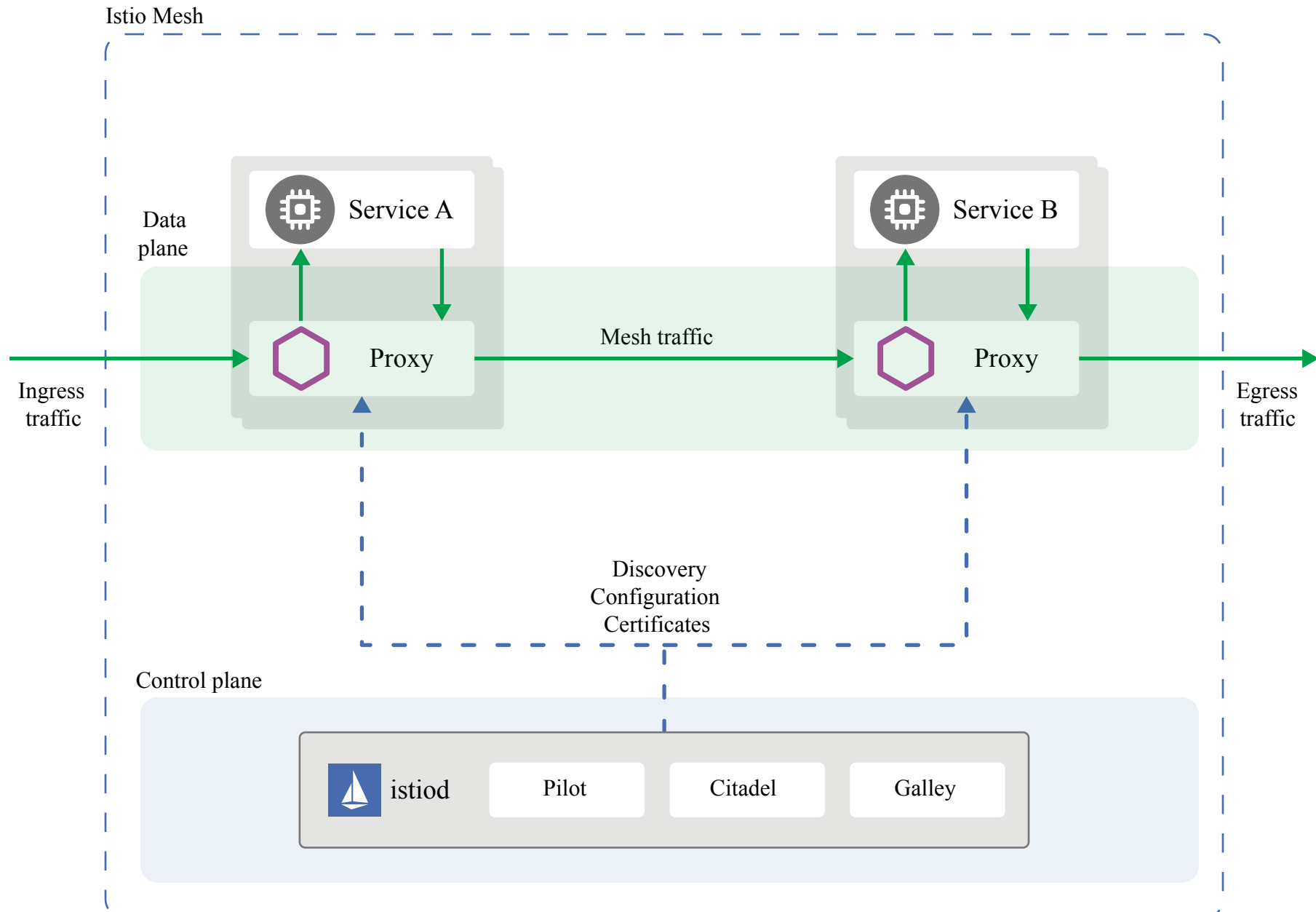


# Proxies

- JupyterHub - CHP (by default).
- Kubeflow - Istio.



# Istio



# JupyterHub UI

Start My Server

## Named Servers


In addition to your default server, you may have additional server(s) with names. This allows you to have more than one server running at the same time.

Server name	URL	Last activity	Actions
<input type="text" value="Name your server"/>	<a href="#">Add New Server</a>		
research	<a href="/user/professor/research">/user/professor/research</a>	a minute ago	<a href="#">stop</a>
teaching		11 minutes ago	<a href="#">start</a> <a href="#">delete</a>

Source:

<https://github.com/jupyterhub/jupyterhub/blob/ed8a531b85b6d64450acfc9235878fca72f7d798/doc/servers-home.png>

# Kubeflow UI

 **Kubeflow**

- Home
- Notebooks**
- Volumes
- Experiments (KFP)
- Pipelines
- Runs
- Recurring Runs
- Experiments (AutoML)
- Models
- Tensorboards

---

Manage Contributors

---

[GitHub](#)

[Documentation](#)

---

Privacy • Usage Reporting  
build version dev\_local

cluster-tests-kf (Owner)

## ← New notebook

Custom Notebook

Image

[Advanced Options](#)

**CPU / RAM** ?

Minimum CPU: 0.5

Minimum Memory Gi: 1

[Advanced Options](#)

**GPUs**

Number of GPUs: None

GPU Vendor

**Workspace Volume**

Volume that will be mounted in your home directory.

New volume -volume, Empty, 10Gi

**Data Volumes**

Additional volumes that will be mounted in your Notebook

[+ Add new volume](#) [+ Attach existing volume](#)

[Advanced Options](#)

**LAUNCH** **CANCEL**

## In JupyterHub's defense

To quote one of the maintainers from the aforementioned issue:

The target use case was a single machine with 5-50 users, and several design decisions were taken with user-space installability, maintainability, and simplicity in mind, while scalability was explicitly out of scope as something we knew we didn't have the resources to tackle.

## Backend agnosticism

```
<pre class="mermaid"> %%{init: {'theme':'dark'}}%% graph TD; JH(JupyterHub) --> DockerSpawner JH --> SudoSpawner JH --> BatchSpawner JH --> YarnSpawner JH --> SSHSpawner JH --> KubeSpawner </pre>
```

## Closing

- Backend agnosticism comes at a price.
- Modular design confers important benefits.
- Scalability and resilience bubble up.
- Avoid the sunk cost fallacy.
- There are no perfect architectural choices.

# Thank you

```
<script type="module"> import mermaid from  
'https://cdn.jsdelivr.net/npm/mermaid@10/dist/mermaid.esm.min.mjs'; mermaid.initialize({  
startOnLoad: true }); window.addEventListener('vscode.markdown.updateContent',  
function() { mermaid.init() }); </script>
```