RECOMMENDATION AND STARFISH

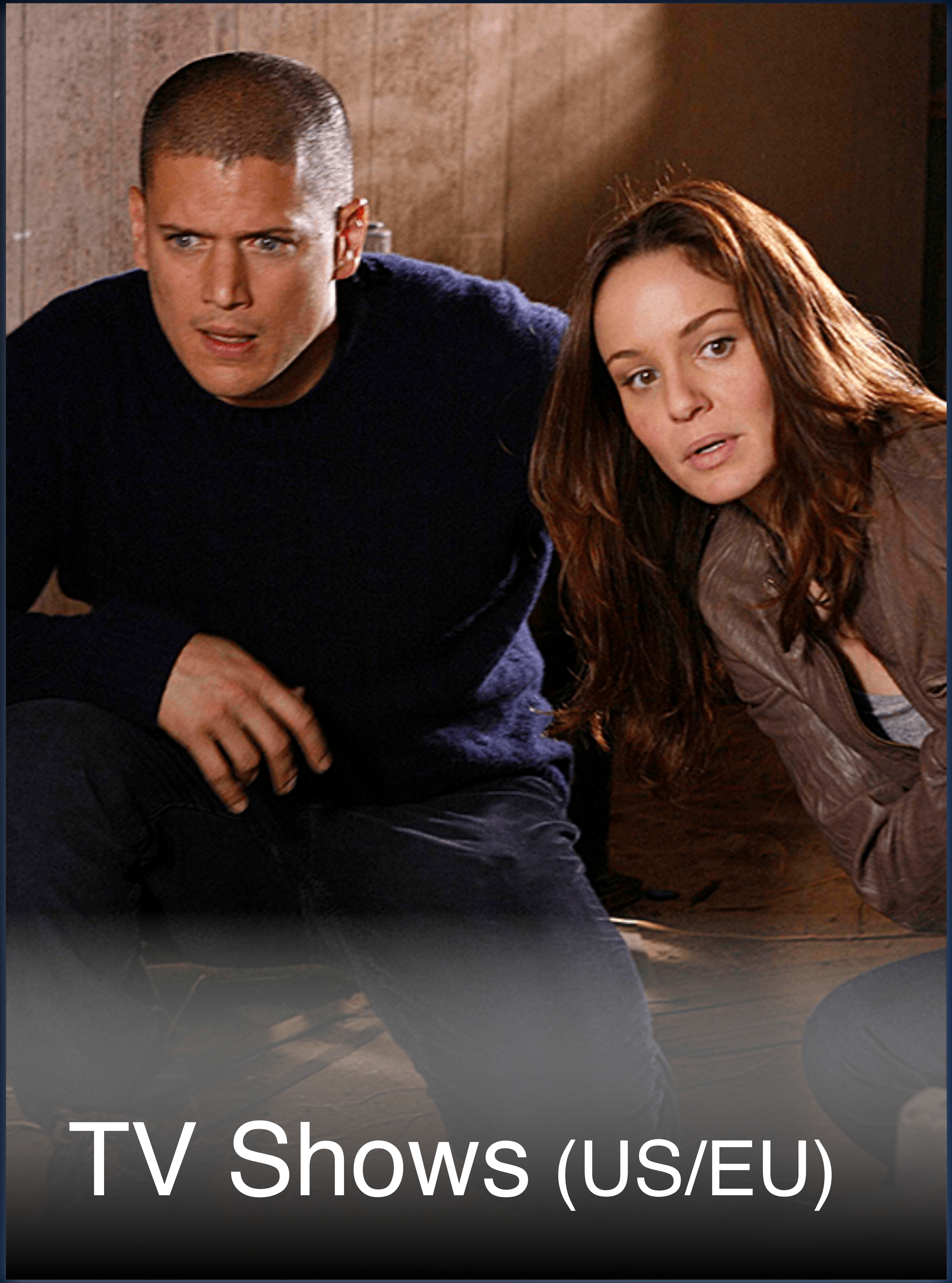
Shripad Deshmukh

U-NEXT

Watch, read, and listen. All of your entertainment in one place.

U-NEXT

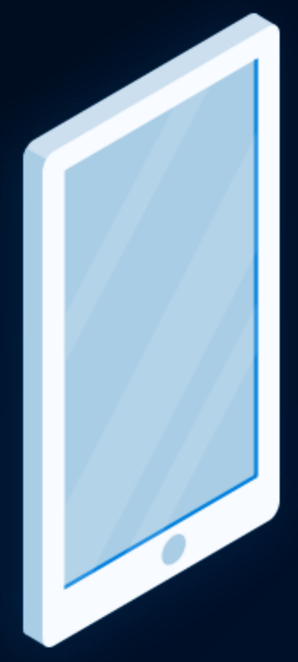
170,000+ films & TV shows



170,000+ films & TV shows

To all platforms

TVs



Smart devices

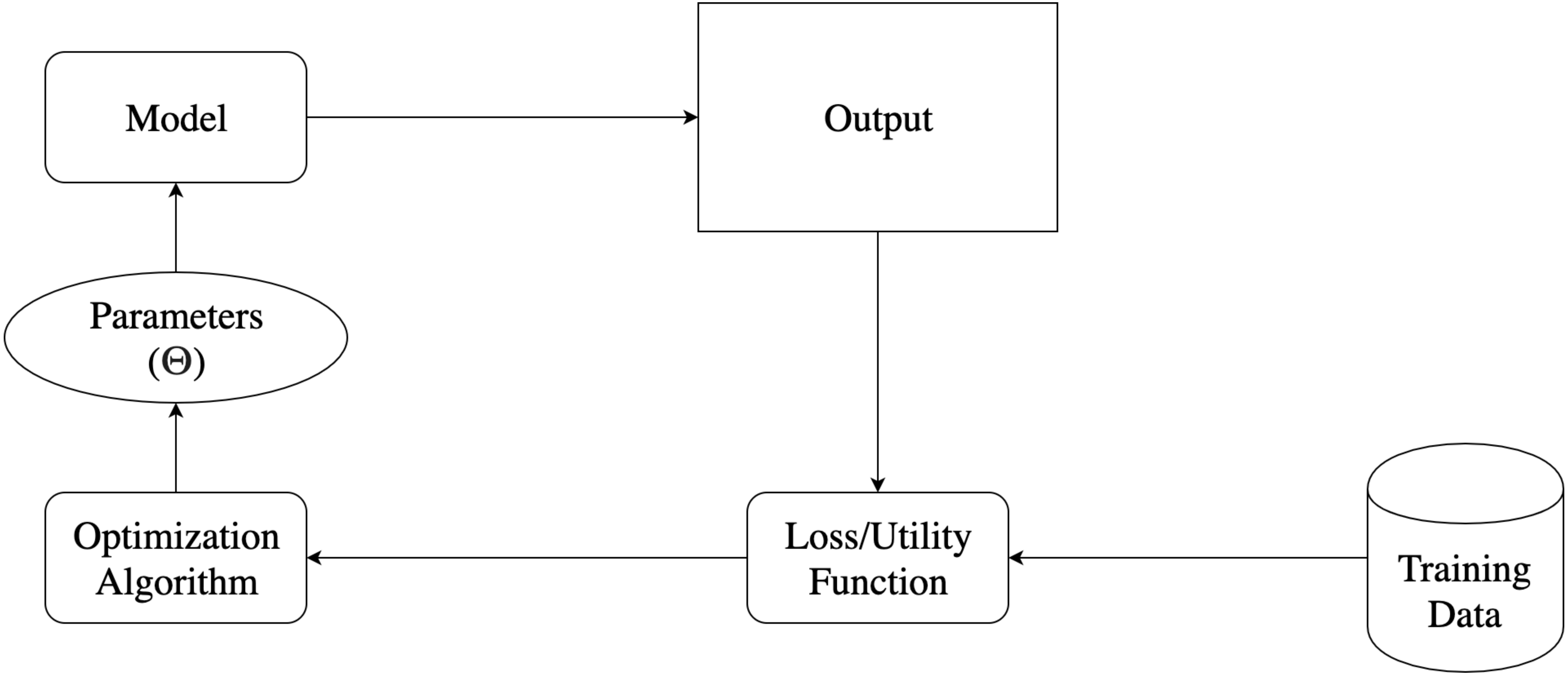
Gaming consoles

PCs

IMPLICIT DATA

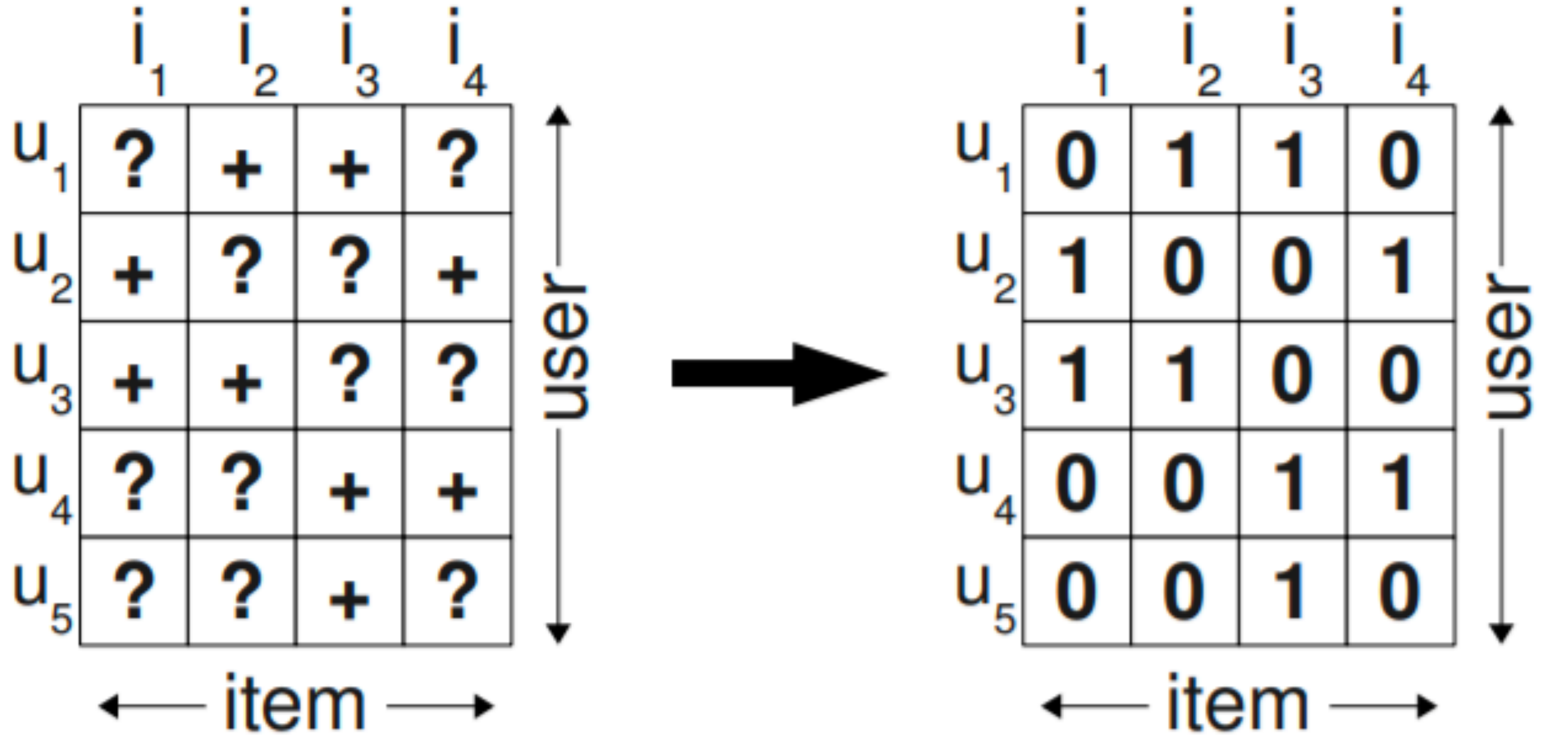
- Implicit Data:
  - No negative feedback
  - Inherently noisy
  - Preference vs Confidence

LEARNING RECOMMENDER SYSTEM



TRAINING DATA

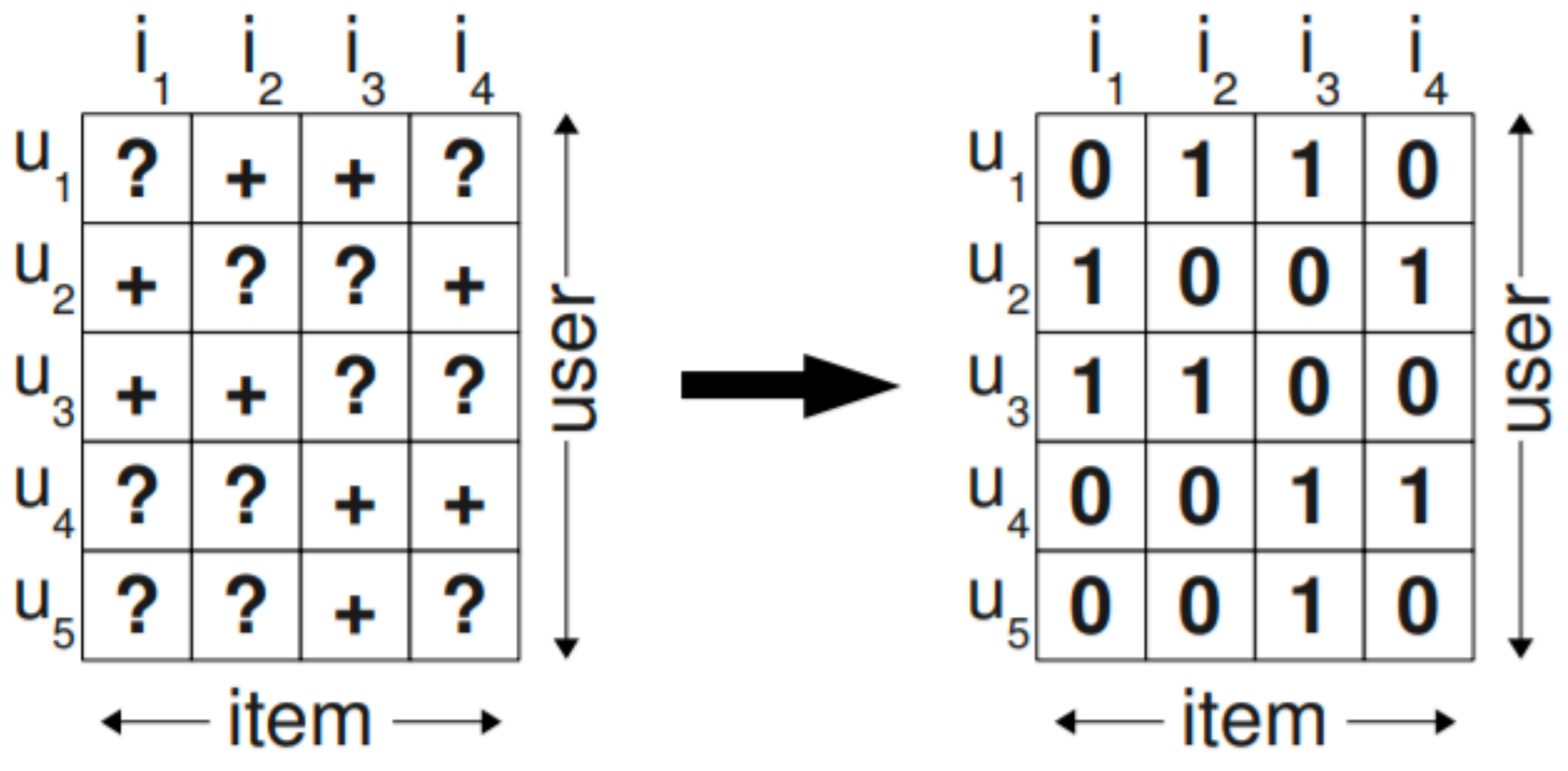
Xui -> Score for an item that reflects
Users preference for that item

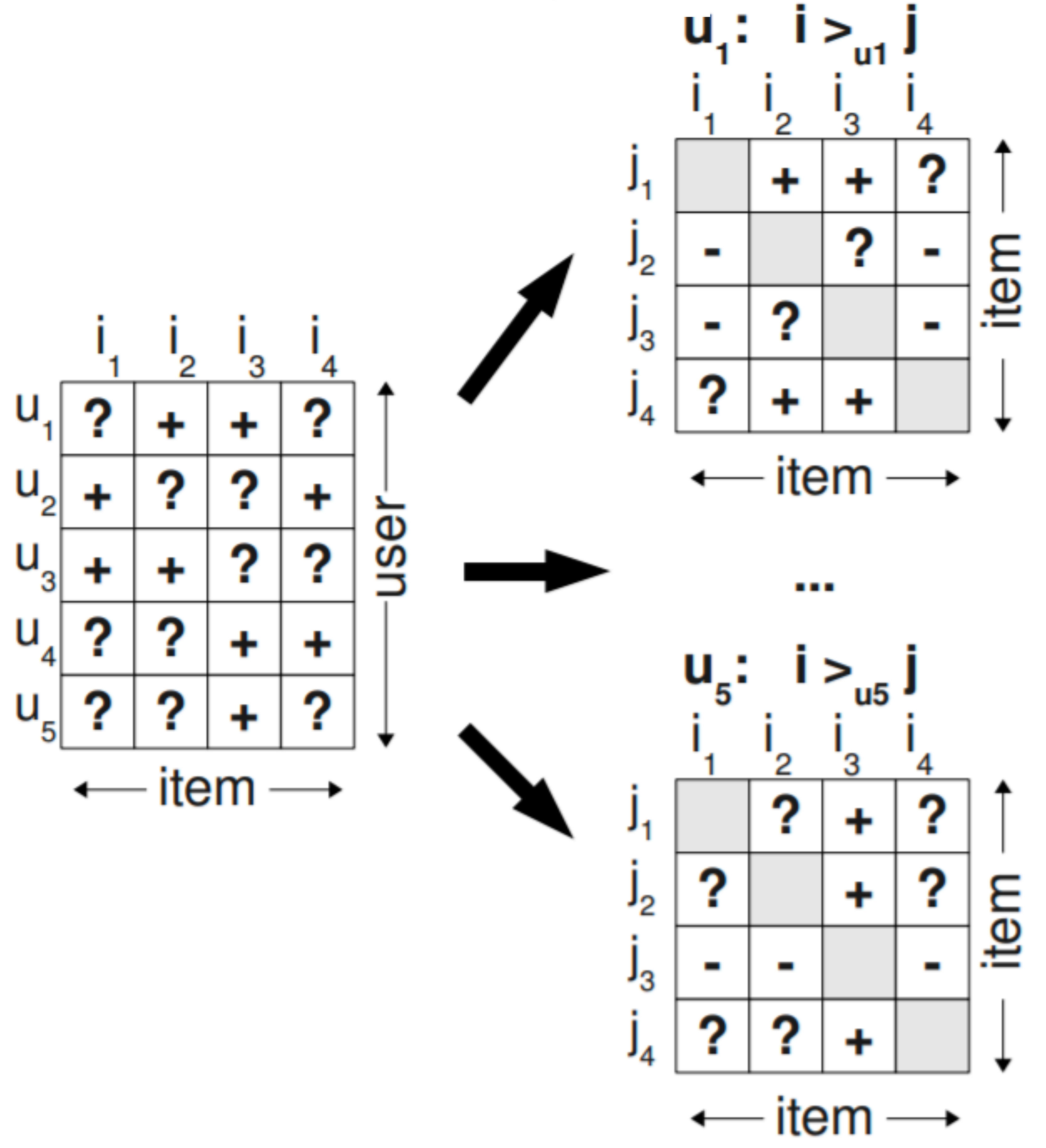


TRAINING DATA

Xui -> Score for an item that reflects
Users preference for that item

$(u, i, j) \in D_S$





BAYESIAN OPT

The Bayesian formulation of finding the correct personalized ranking for all items $i \in I$ is to maximize the following posterior probability where Θ represents the parameter vector of an arbitrary model class (e.g. matrix factorization).

$$p(\Theta | >_u) \propto p(>_u | \Theta)\, p(\Theta)$$

$>u$ is the desired but latent preference structure for user u
$p(>u \mid \theta)$ is a user specific likelihood function

BAYESIAN OPT

The Bayesian formulation of finding the correct personalized ranking for all items $i \in I$ is to maximize the following posterior probability where Θ represents the parameter vector of an arbitrary model class (e.g. matrix factorization).

$$p(\Theta| >_u) \propto p(>_u |\Theta)\, p(\Theta)$$

$>u$ is the desired but latent preference structure for user u
$p(>u \mid \theta)$ is a user specific likelihood function

All users are presumed to act independently of each other. We also assume the ordering of each pair of items (i, j) for a specific user is independent of the ordering of every other pair, we can formulate the individual probability that a user prefers item i to item j as follows

$$p(i >_u j|\Theta) := \sigma(\hat{x}_{uij}(\Theta))$$

where σ is the logistic sigmoid:

$$\sigma(x) := \frac{1}{1+e^{-x}}$$

$$p(\Theta) \sim N(0, \Sigma_\Theta)$$

BAYESIAN OPT

The Bayesian formulation of finding the correct personalized ranking for all items i ∈ I is to maximize the following posterior probability where Θ represents the parameter vector of an arbitrary model class (e.g. matrix factorization).

$$p(\Theta| >_u) \propto p(>_u |\Theta)\, p(\Theta)$$

>u is the desired but latent preference structure for user u
p(>u | θ) is a user specific likelihood function

All users are presumed to act independently of each other. We also assume the ordering of each pair of items (i, j) for a specific user is independent of the ordering of every other pair, we can formulate the individual probability that a user prefers item i to item j as follows

$$p(i >_u j|\Theta) := \sigma(\hat{x}_{uij}(\Theta))$$

where σ is the logistic sigmoid:

$$\sigma(x) := \frac{1}{1+e^{-x}}$$

$$p(\Theta) \sim N(0, \Sigma_\Theta)$$

BPR-Opt := ln p(Θ| >u)

$$= \sum_{(u,i,j)\in D_S} \ln \sigma(\hat{x}_{uij}) - \lambda_\Theta ||\Theta||^2$$

where λΘ are model specific regularization parameters.

ANALOGIES TO AUC OPTIMIZATION

$$\text{BPR-O}\textsc{pt} := \sum_{(u,i,j)\in D_S} \ln \sigma(\hat{x}_{uij}) - \lambda_\Theta ||\Theta||^2$$

$$\text{AUC}(u) = \sum_{(u,i,j)\in D_S} z_u \, \delta(\hat{x}_{uij} > 0)$$

The analogy between BPR-Opt and (1) is obvious. Besides the normalizing constant zu they only differ in the loss function.
The AUC uses the non-differentiable loss δ(x > 0) which is identical to the Heaviside function:

$$\delta(x > 0) = H(x) := \begin{cases} 1, & x > 0 \\ 0, & \text{else} \end{cases}$$

Optimizing for BPR-OPT criterion will be similar to optimizing for AUC which is a rank based metric as we can see from the above equation.

STOCHASTIC GRADIENT DESCENT USING BOOTSTRAPPING

$$\frac{\partial \text{BPR-Opt}}{\partial \Theta} = \sum_{(u,i,j) \in D_S} \frac{\partial}{\partial \Theta} \ln \sigma(\hat{x}_{uij}) - \lambda_\Theta \frac{\partial}{\partial \Theta} ||\Theta||^2$$

$$\propto \sum_{(u,i,j) \in D_S} \frac{-e^{-\hat{x}_{uij}}}{1 + e^{-\hat{x}_{uij}}} \cdot \frac{\partial}{\partial \Theta} \hat{x}_{uij} - \lambda_\Theta \Theta$$

$$\Theta \leftarrow \Theta + \alpha \left(\frac{e^{-\hat{x}_{uij}}}{1 + e^{-\hat{x}_{uij}}} \cdot \frac{\partial}{\partial \Theta} \hat{x}_{uij} + \lambda_\Theta \Theta \right)$$

STOCHASTIC GRADIENT DESCENT USING BOOTSTRAPPING

$$\frac{\partial \text{BPR-Opt}}{\partial \Theta} = \sum_{(u,i,j)\in D_S} \frac{\partial}{\partial \Theta} \ln \sigma(\hat{x}_{uij}) - \lambda_\Theta \frac{\partial}{\partial \Theta} ||\Theta||^2$$

$$\propto \sum_{(u,i,j)\in D_S} \frac{-e^{-\hat{x}_{uij}}}{1+e^{-\hat{x}_{uij}}} \cdot \frac{\partial}{\partial \Theta}\hat{x}_{uij} - \lambda_\Theta \Theta$$

$$\Theta \leftarrow \Theta + \alpha \left(\frac{e^{-\hat{x}_{uij}}}{1+e^{-\hat{x}_{uij}}} \cdot \frac{\partial}{\partial \Theta}\hat{x}_{uij} + \lambda_\Theta \Theta \right)$$

procedure LearnBPR(D_S, Θ)
  initialize Θ
  repeat
    draw (u, i, j) from D_S
    $\Theta \leftarrow \Theta + \alpha \left(\frac{e^{-\hat{x}_{uij}}}{1+e^{-\hat{x}_{uij}}} \cdot \frac{\partial}{\partial \Theta}\hat{x}_{uij} + \lambda_\Theta \cdot \Theta \right)$
  until convergence
  return $\hat{\Theta}$
end procedure

LEARNING MODELS WITH BPR

Matrix Factorization and learned k-nearest-neigbhor both try to model the hidden preferences of a user on an item but Matrix factorization shows better result.

$$\hat{x}_{uij} := \hat{x}_{ui} - \hat{x}_{uj}$$

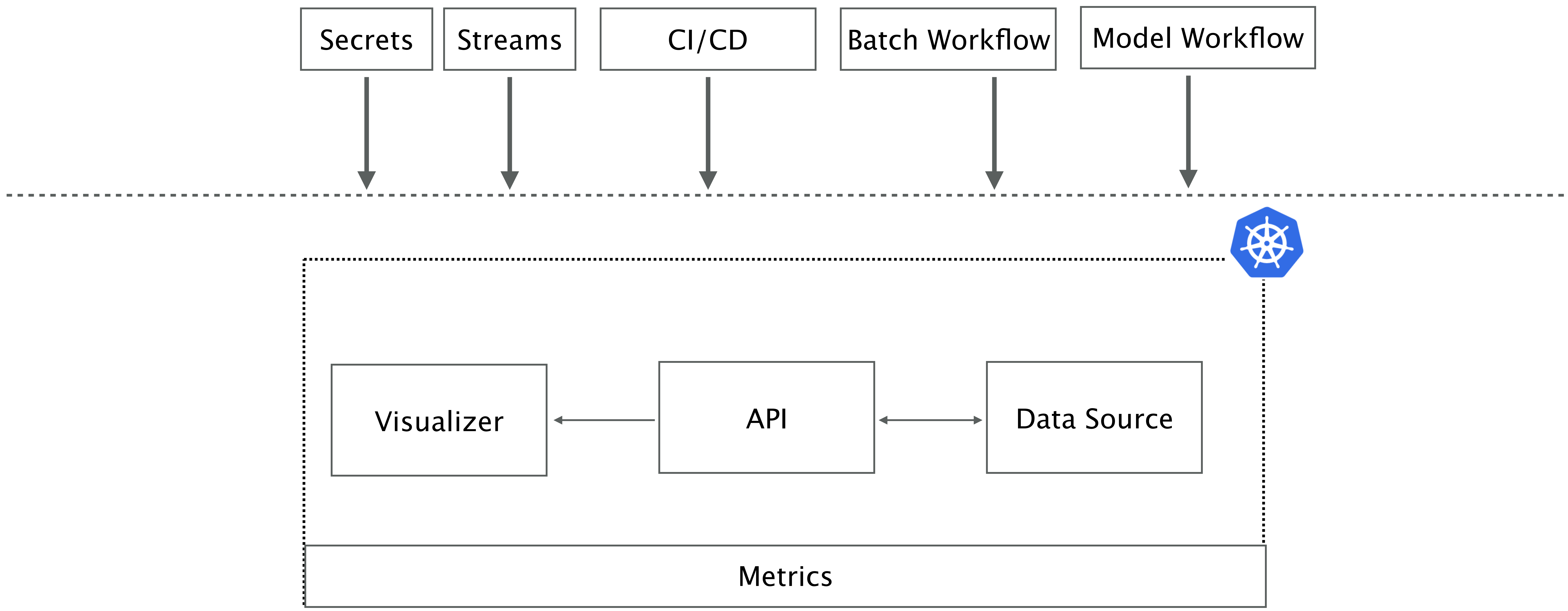
PAIN POINTS

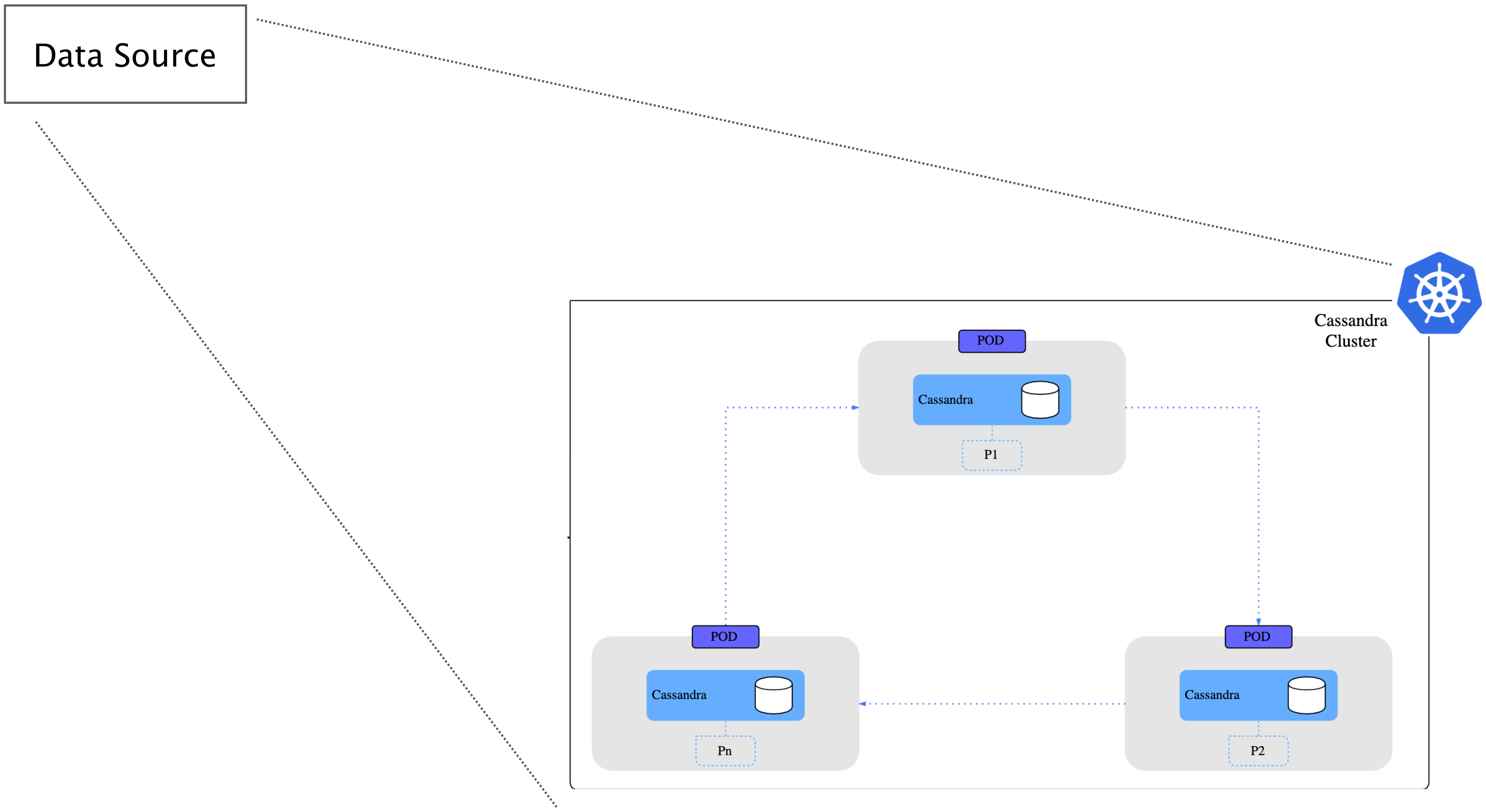
End-to-end orchestration of idea to production

- Easier experimentation
- Scaling backend computation
- Serving precomputed data
- Serving model in real-time

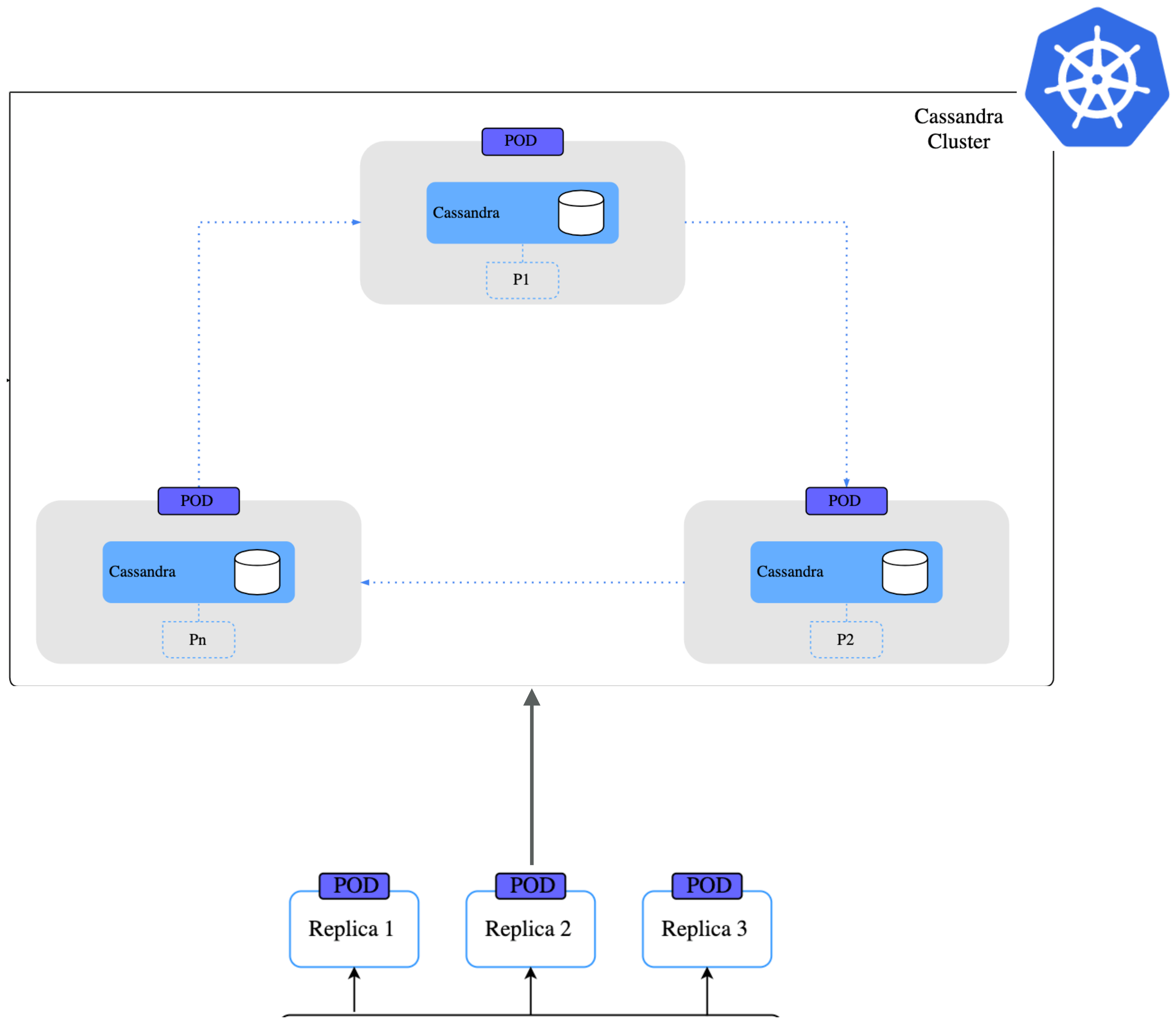
STARFISH

- Platform of Data Science services
- Supports;
  - Precomputed big data
  - Support realtime models
  - A/B

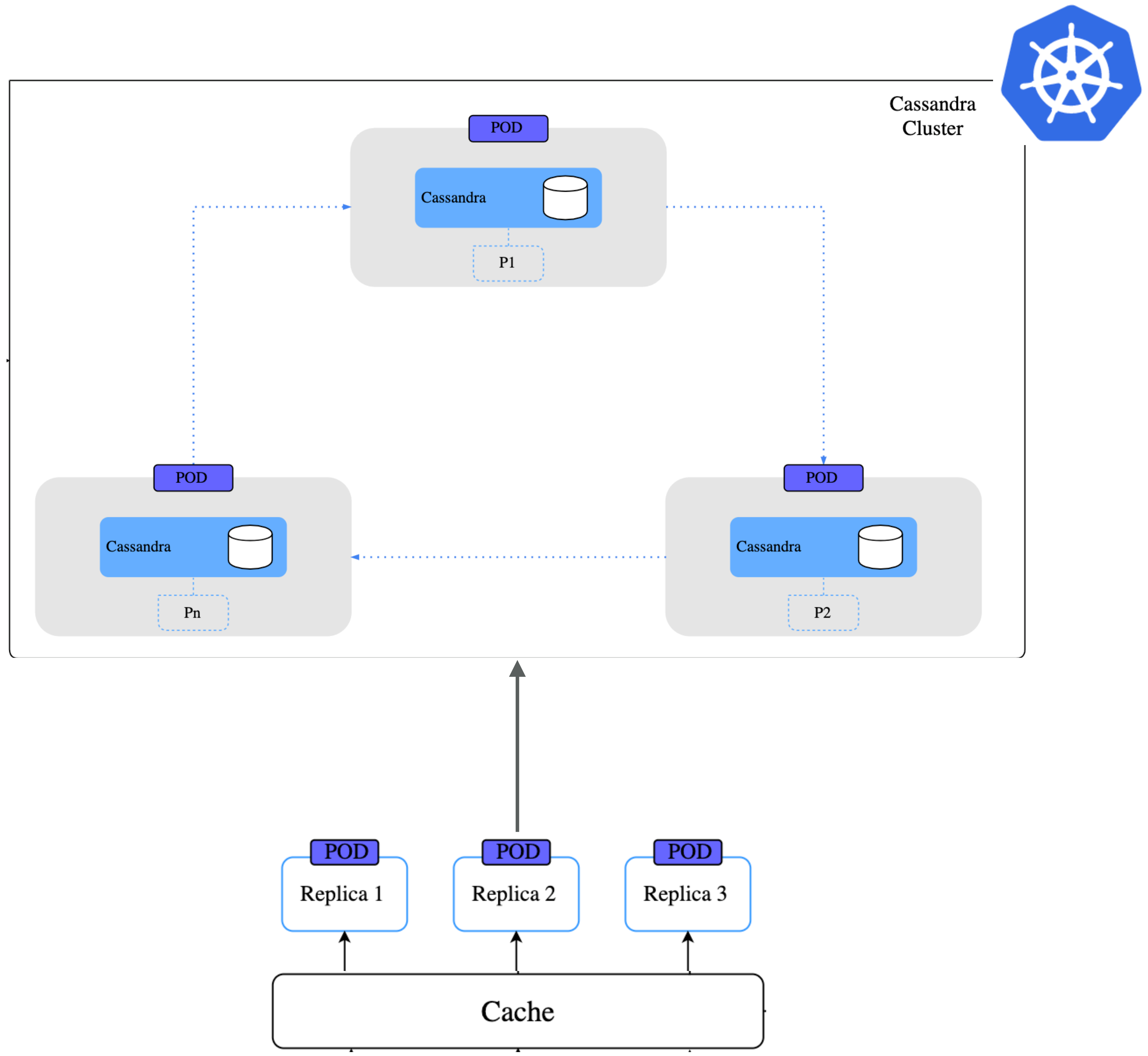

Secrets
Streams
CI/CD
Batch Workflow
Model Workflow
Visualizer
API
Data Source
Metrics


Data Source
Cassandra Cluster
POD
Cassandra
P1
POD
Cassandra
Pn
POD
Cassandra
P2

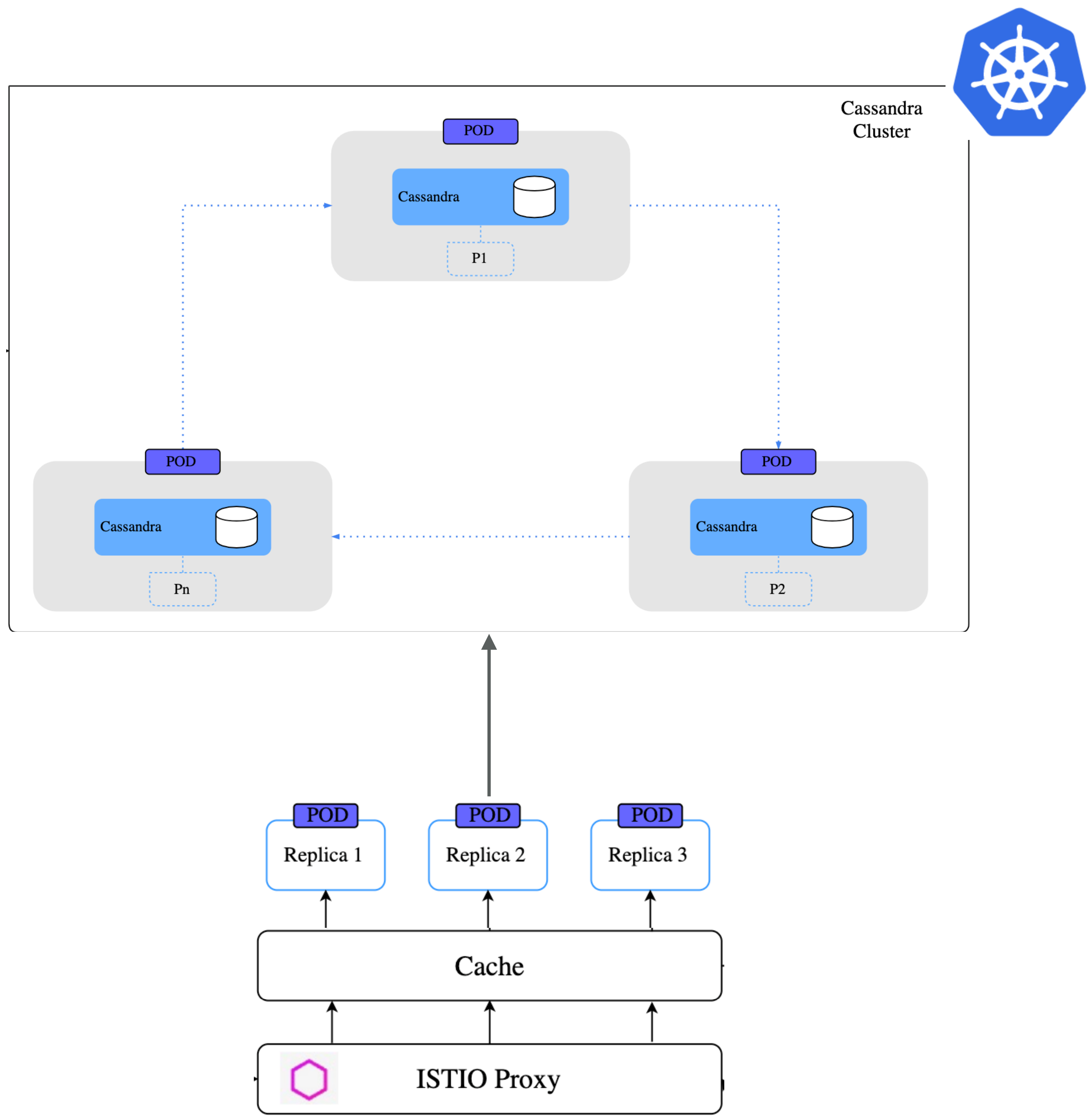

API
Data Source


Cassandra
Cluster
POD
Cassandra
P1
POD
Cassandra
Pn
POD
Cassandra
P2
POD
Replica 1
POD
Replica 2
POD
Replica 3

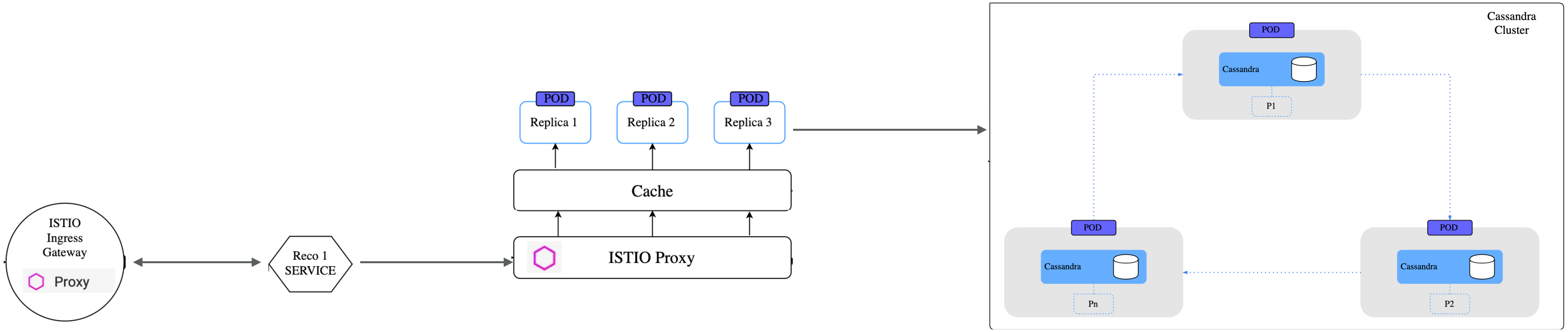

API
Data Source

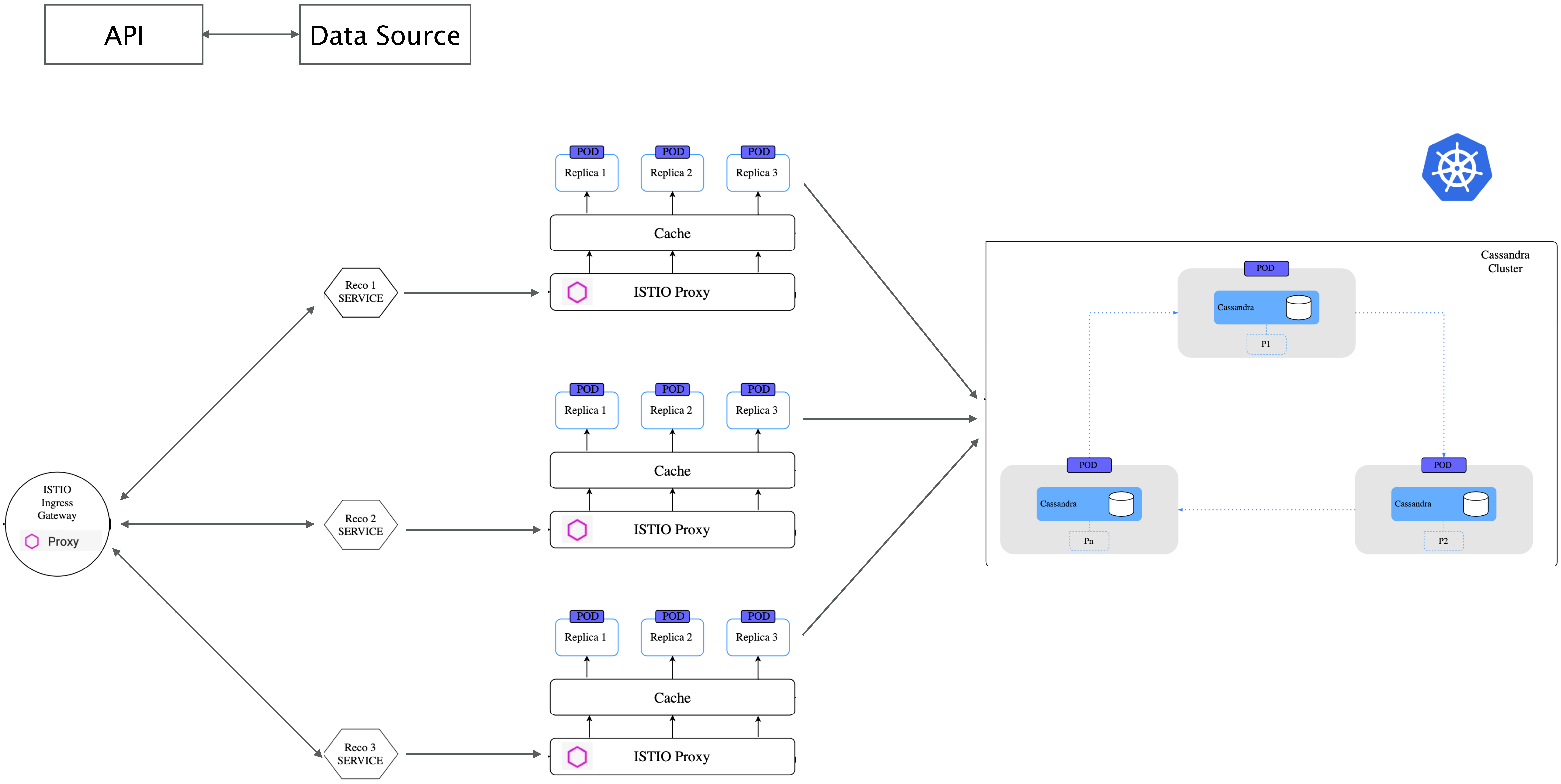

Cassandra
Cluster
POD
Cassandra
P1
POD
Cassandra
Pn
POD
Cassandra
P2
POD
Replica 1
POD
Replica 2
POD
Replica 3
Cache


API
Data Source


Cassandra
Cluster
POD
Cassandra
P1
POD
Cassandra
Pn
POD
Cassandra
P2
POD
Replica 1
POD
Replica 2
POD
Replica 3
Cache
ISTIO Proxy

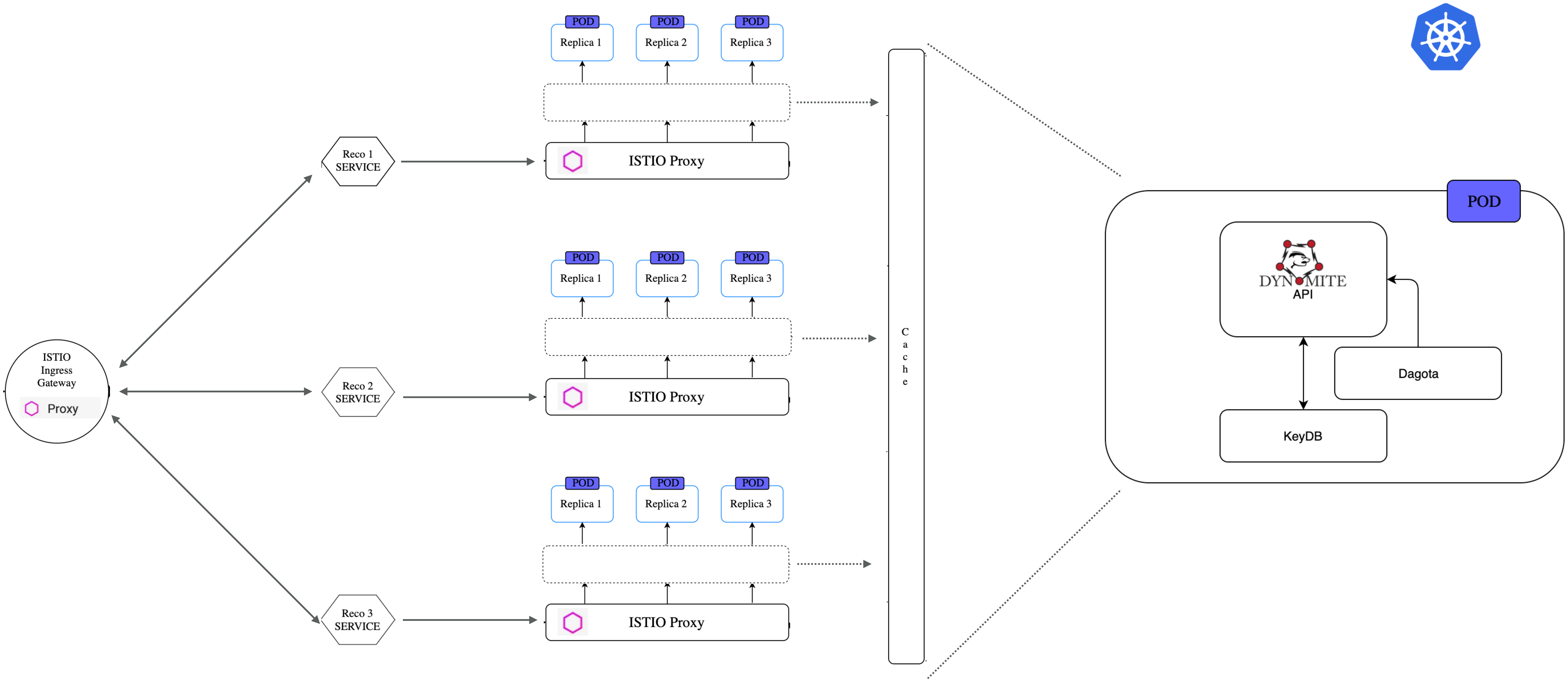

API
Data Source


ISTIO
Ingress
Gateway
Proxy
Reco 1
SERVICE
ISTIO Proxy
Cache
POD
Replica 1
POD
Replica 2
POD
Replica 3
Cassandra
Cluster
POD
Cassandra
P1
POD
Cassandra
Pn
POD
Cassandra
P2

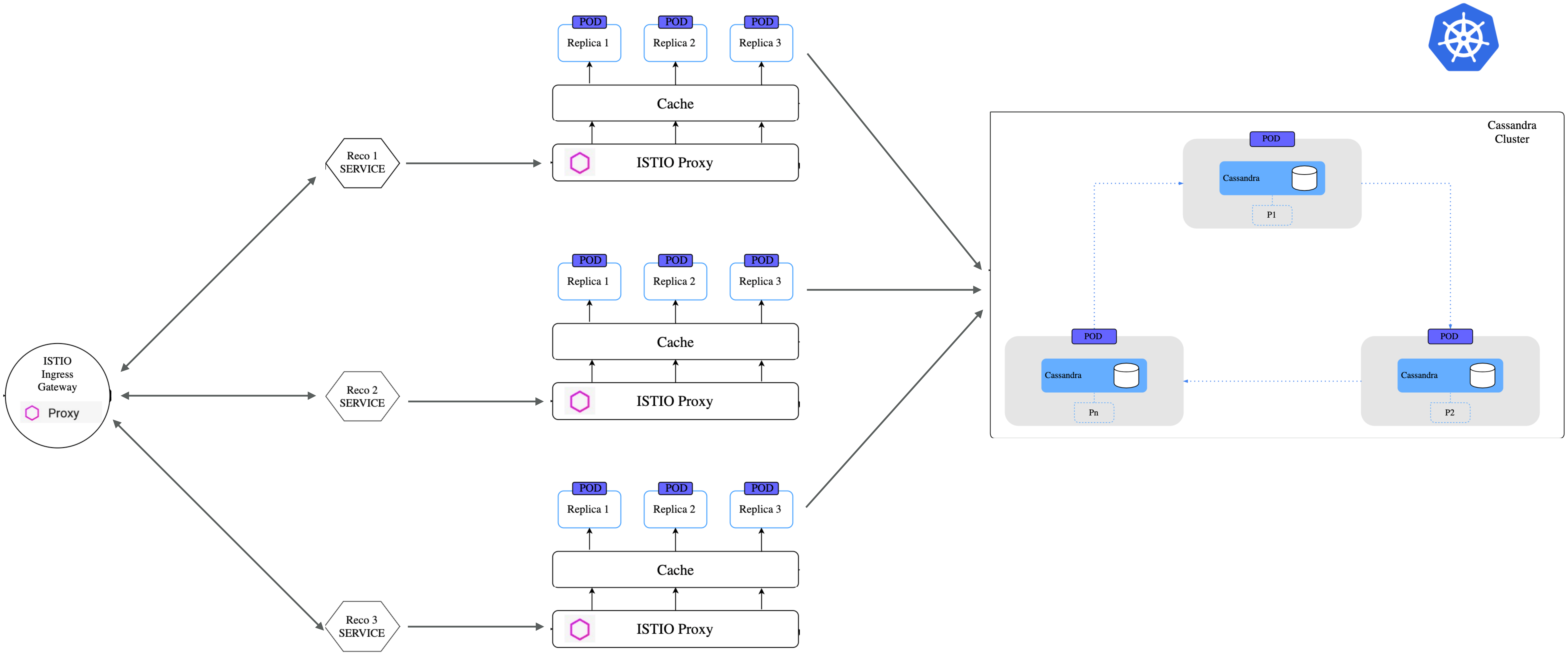


API
Data Source
POD
Replica 1
POD
Replica 2
POD
Replica 3
Cache
ISTIO Proxy
Reco 1 SERVICE
Reco 2 SERVICE
Reco 3 SERVICE
ISTIO Ingress Gateway
Proxy
Cassandra Cluster
Cassandra
P1
Pn
P2

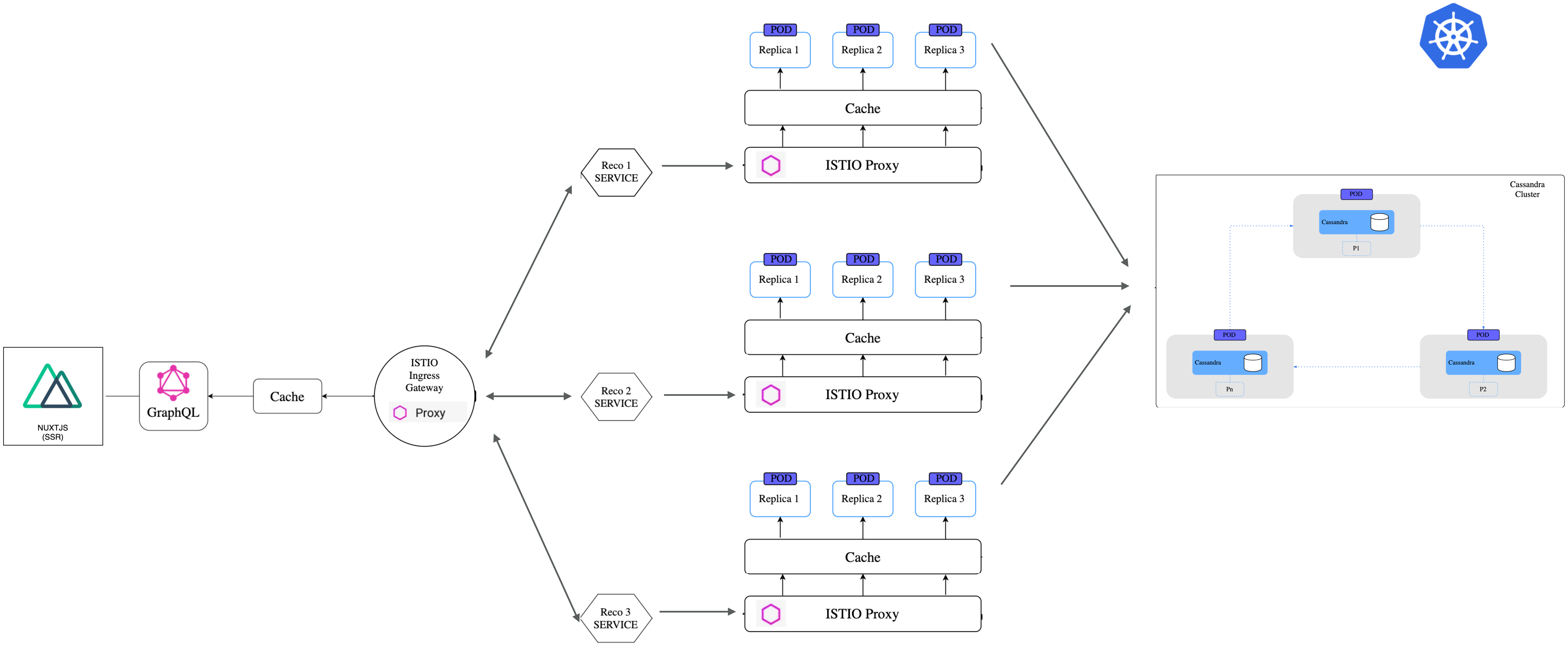

API
Data Source


POD
Replica 1
POD
Replica 2
POD
Replica 3
Reco 1
SERVICE
ISTIO Proxy
ISTIO
Ingress
Gateway
Proxy
POD
Replica 1
POD
Replica 2
POD
Replica 3
Reco 2
SERVICE
ISTIO Proxy
POD
Replica 1
POD
Replica 2
POD
Replica 3
Reco 3
SERVICE
ISTIO Proxy
Cache
POD
DYNOMITE
API
Dagota
KeyDB


API
Data Source


ISTIO
Ingress
Gateway
Proxy
Reco 1
SERVICE
Reco 2
SERVICE
Reco 3
SERVICE
POD
Replica 1
POD
Replica 2
POD
Replica 3
Cache
ISTIO Proxy
POD
Replica 1
POD
Replica 2
POD
Replica 3
Cache
ISTIO Proxy
POD
Replica 1
POD
Replica 2
POD
Replica 3
Cache
ISTIO Proxy
Cassandra
Cluster
POD
Cassandra
P1
POD
Cassandra
Pn
POD
Cassandra
P2

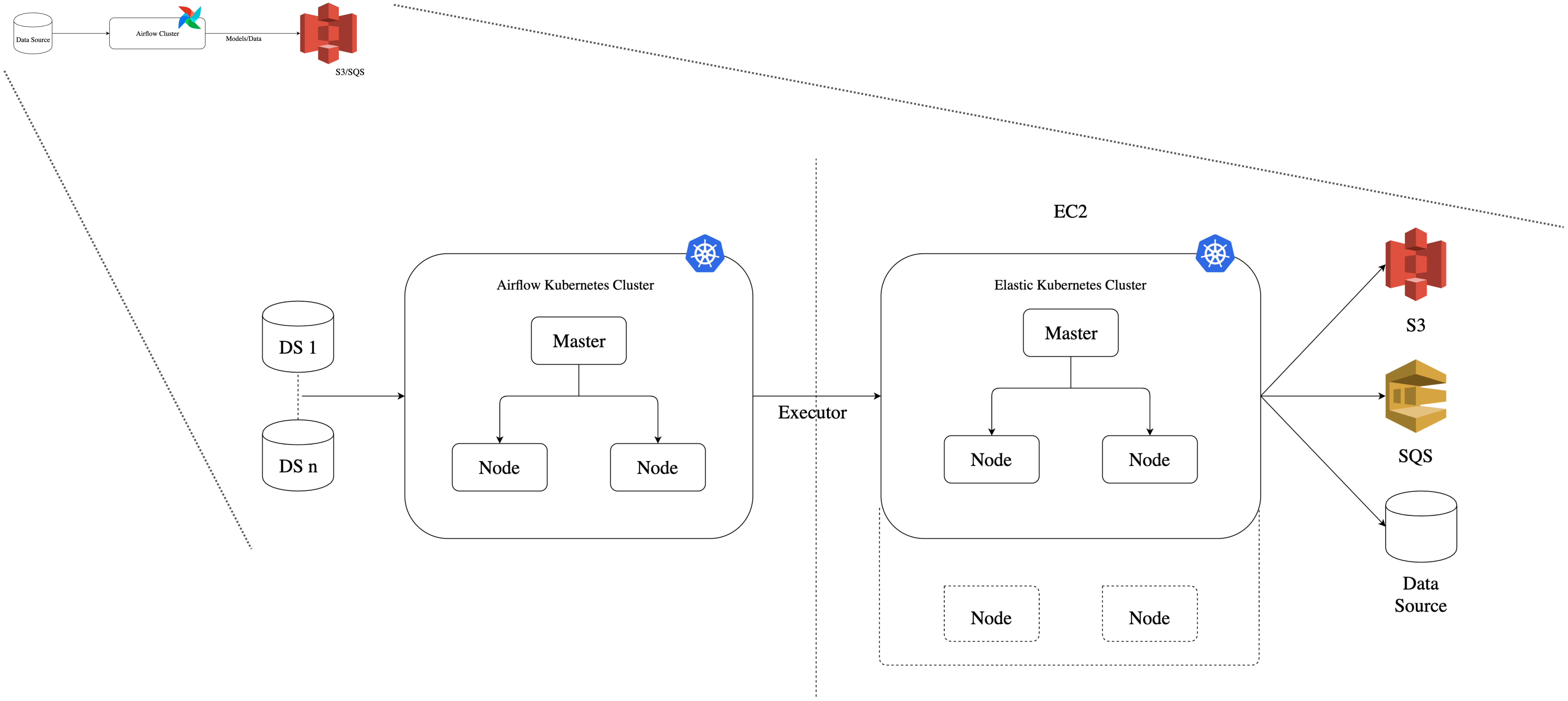

Visualizer
API
Data Source


NUXTJS
(SSR)
GraphQL
Cache
ISTIO
Ingress
Gateway
Proxy
Reco 1
SERVICE
Reco 2
SERVICE
Reco 3
SERVICE
POD
Replica 1
POD
Replica 2
POD
Replica 3
Cache
ISTIO Proxy
Cassandra
Cluster
POD
Cassandra
P1
Pn
P2

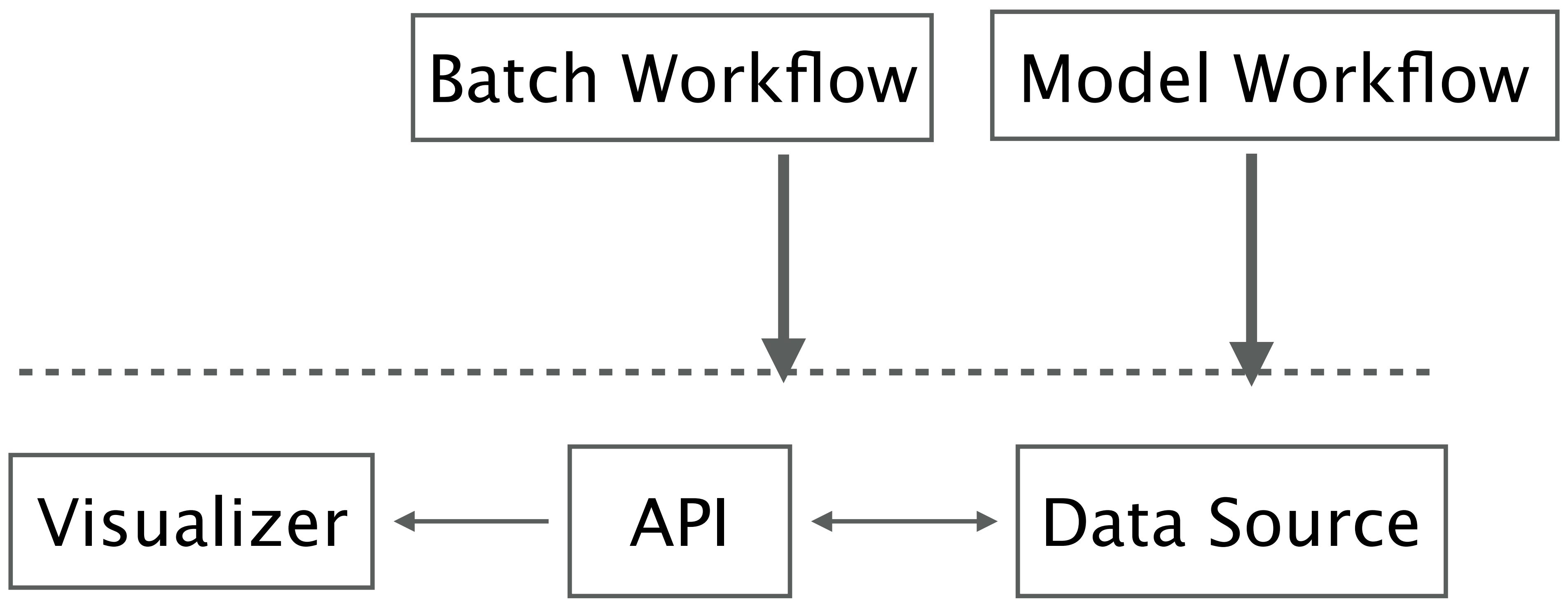

Batch Workflow
Model Workflow
Visualizer
API
Data Source
Data Source
Airflow Cluster
Models/Data
S3/SQS
POD
Replica 1
Replica 2
Replica 3
Cache
ISTIO Proxy
Reco 1 SERVICE
Reco 2 SERVICE
Reco 3 SERVICE
ISTIO Ingress Gateway
Proxy
Cache
GraphQL
NUXTJS (SSR)
Cassandra Cluster
Cassandra
P1
P2
Pn

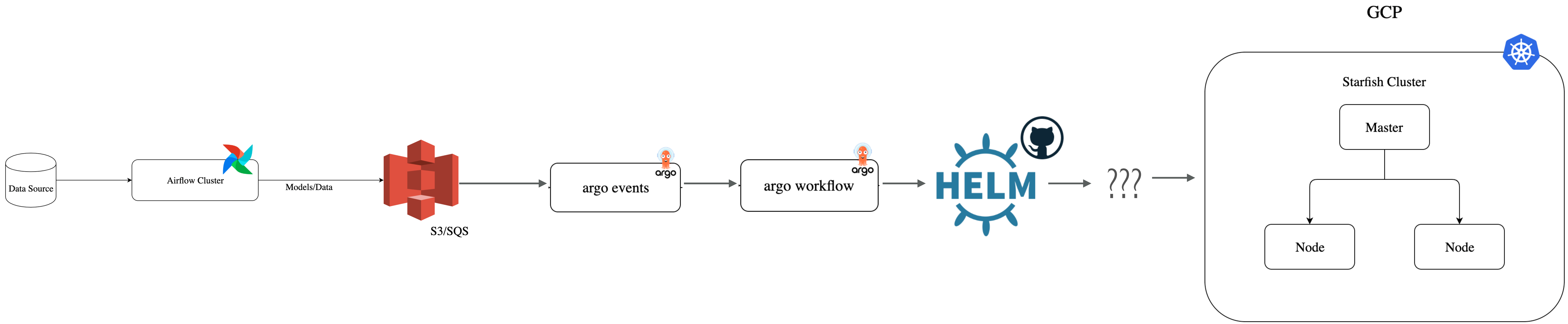
Batch Workflow

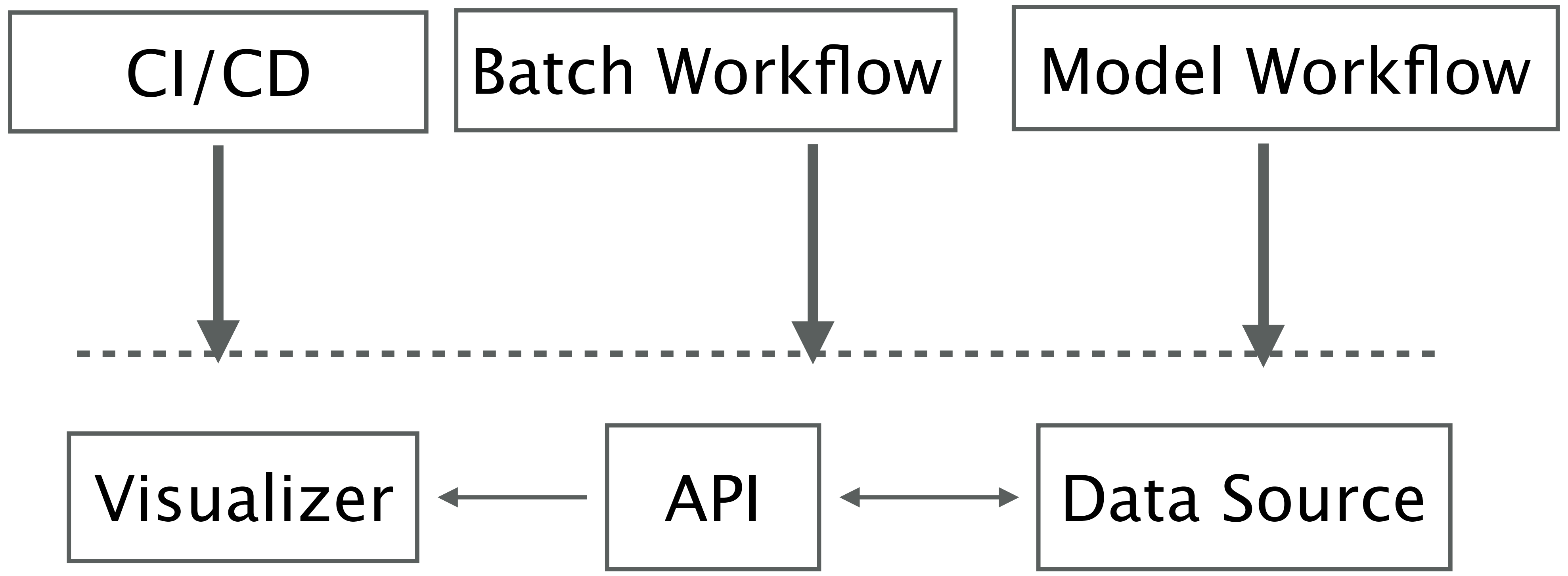
Model Workflow

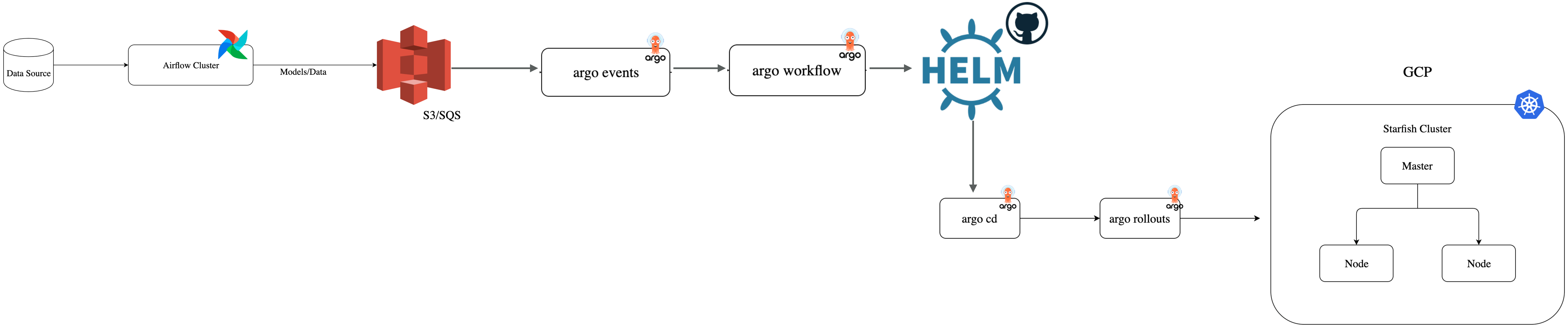


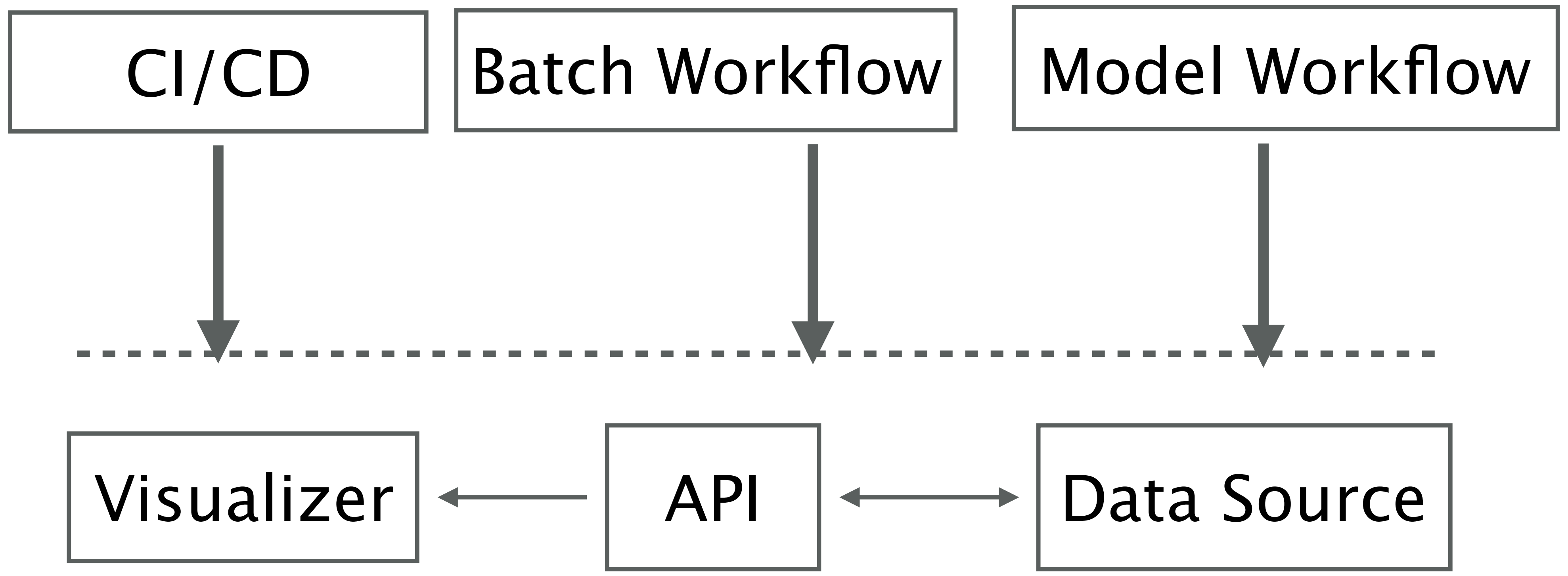

Batch Workflow
Model Workflow
Visualizer
API
Data Source
Data Source
Airflow Cluster
Models/Data
S3/SQS
HELM
argo events
argo workflow
Cassandra Cluster
POD
Cassandra
P1
Pn
P2
POD
Replica 1
Replica 2
Replica 3
Cache
ISTIO Proxy
Reco 1 SERVICE
Reco 2 SERVICE
Reco 3 SERVICE
ISTIO Ingress Gateway
Proxy
Cache
GraphQL
NUXTJS (SSR)

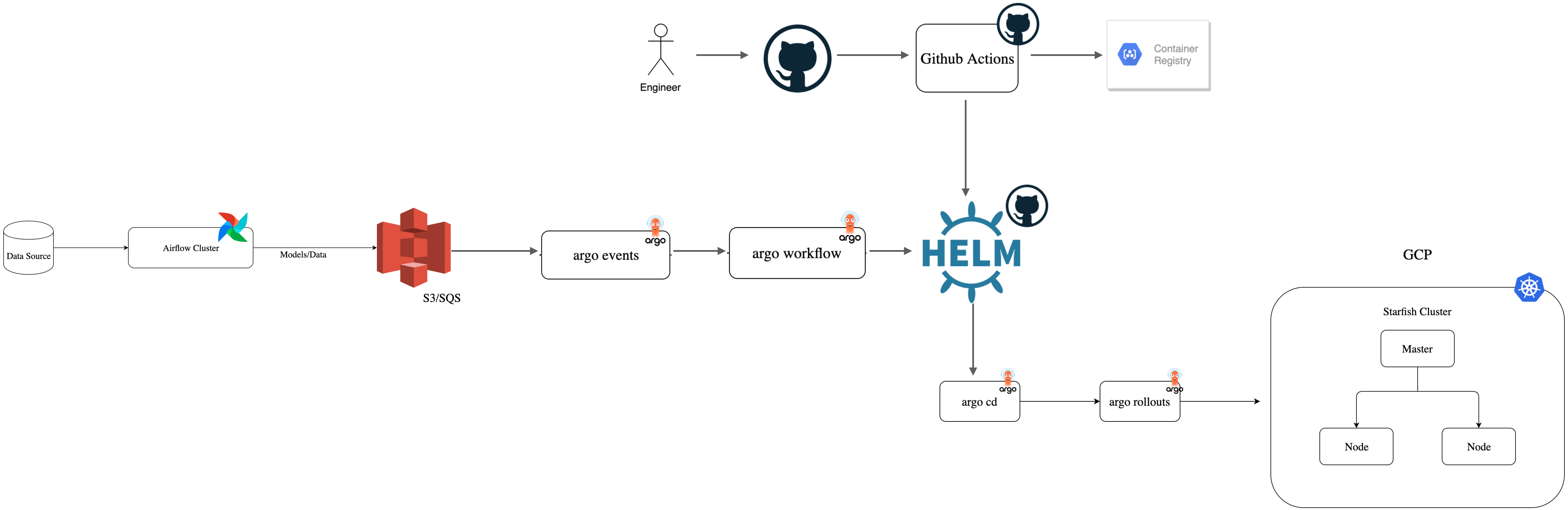

Batch Workflow
Model Workflow
Visualizer
API
Data Source

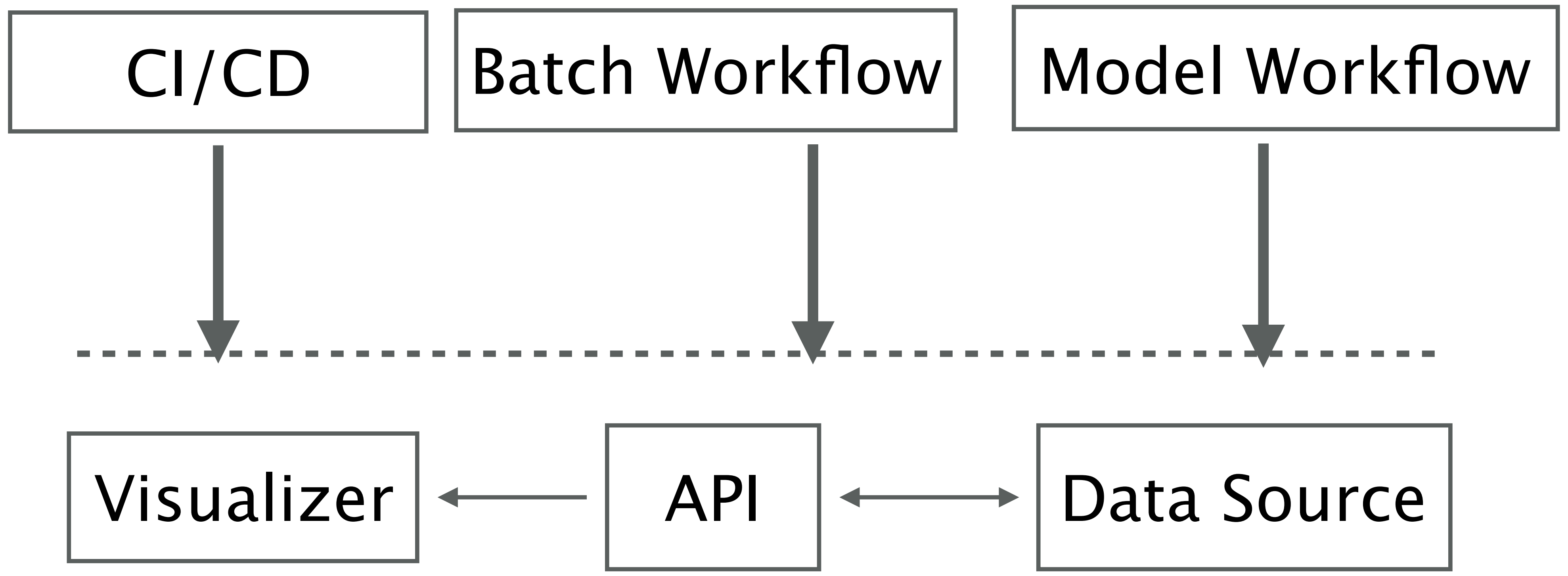

Data Source
Airflow Cluster
Models/Data
S3/SQS
argo events
argo
argo workflow
argo
HELM
???
GCP
Starfish Cluster
Master
Node
Node

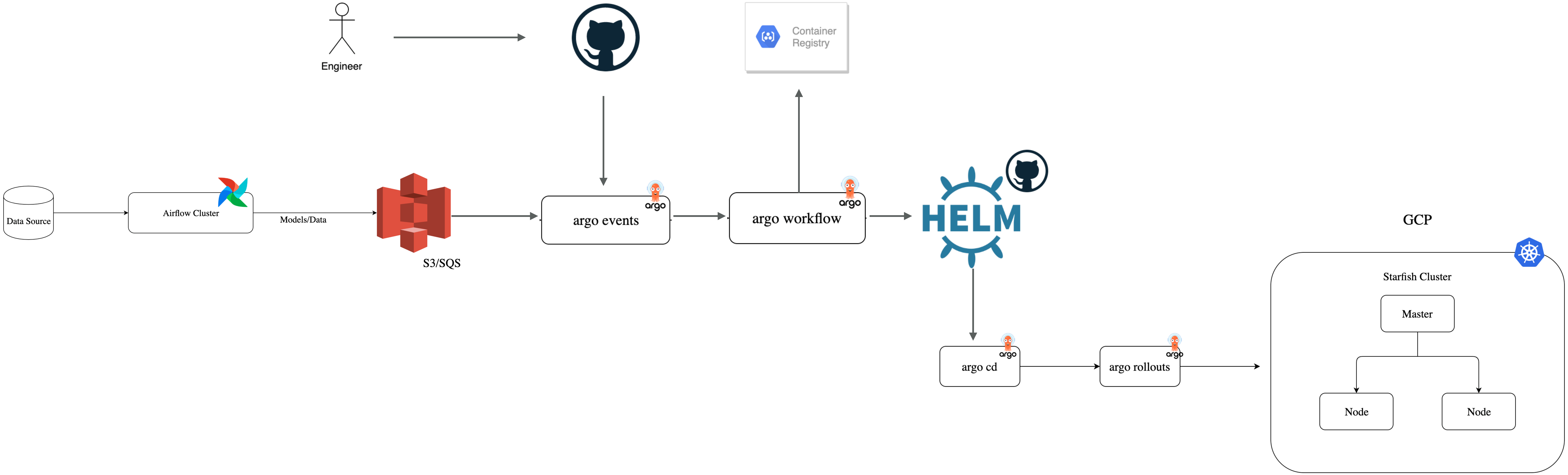

CI/CD
Batch Workflow
Model Workflow
Visualizer
API
Data Source

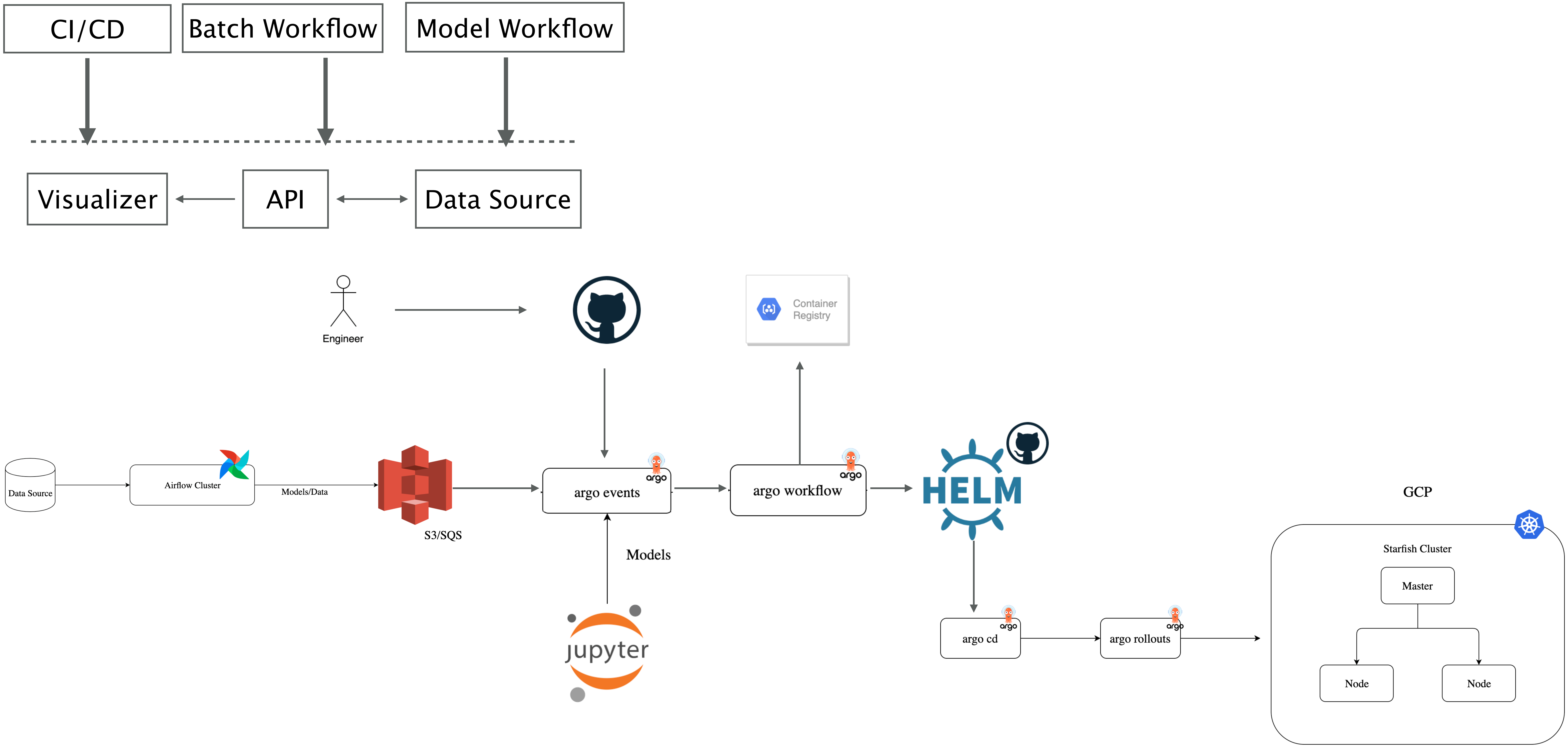

Data Source
Airflow Cluster
Models/Data
S3/SQS
argo events
argo workflow
HELM
argo cd
argo rollouts
GCP
Starfish Cluster
Master
Node
Node

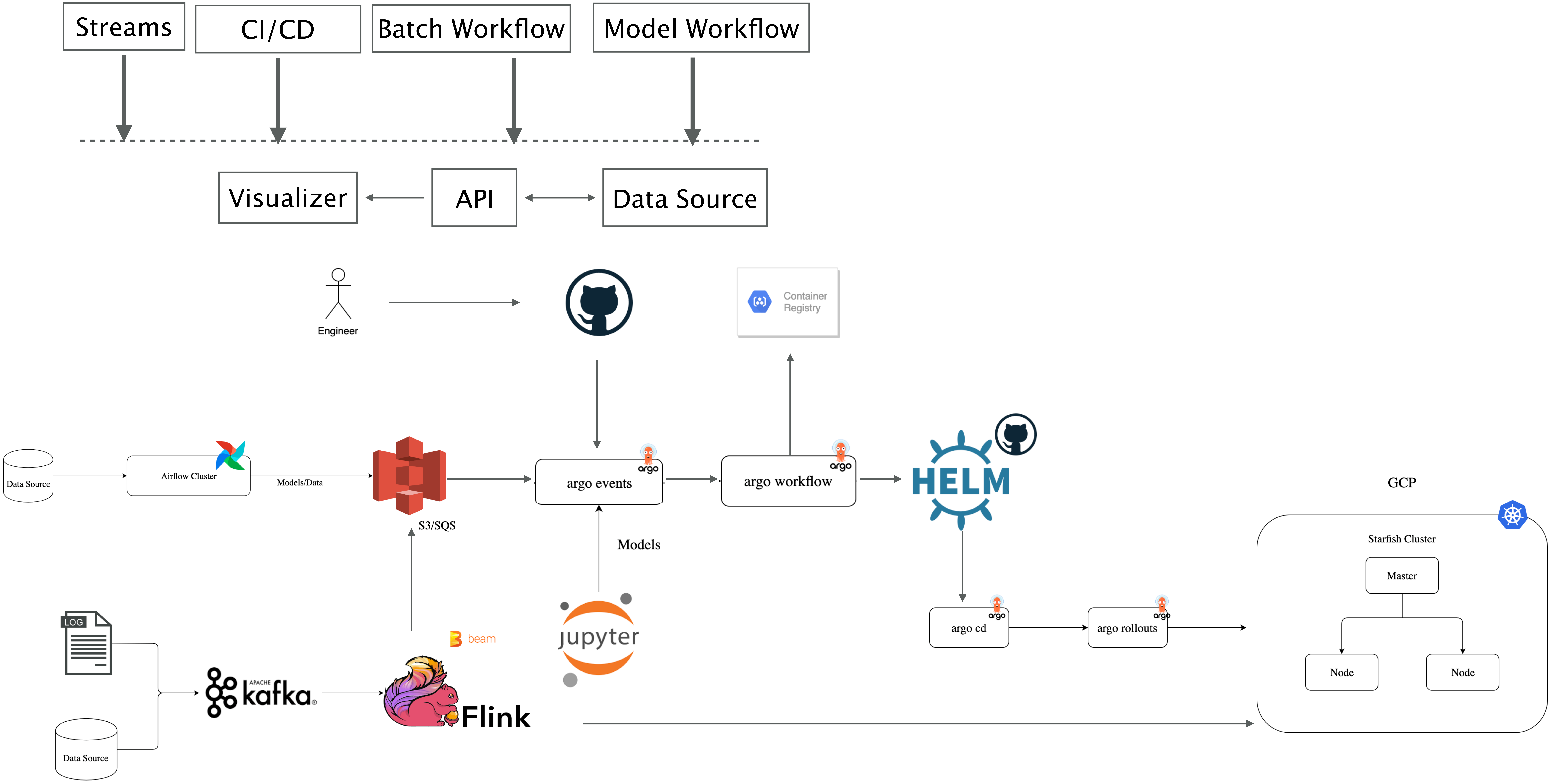

CI/CD
Batch Workflow
Model Workflow
Visualizer
API
Data Source


Engineer
Github Actions
Container
Registry
Data Source
Airflow Cluster
Models/Data
S3/SQS
argo events
argo workflow
HELM
argo cd
argo rollouts
GCP
Starfish Cluster
Master
Node
Node


CI/CD
Batch Workflow
Model Workflow
Visualizer
API
Data Source


Engineer
Container
Registry
Data Source
Airflow Cluster
Models/Data
S3/SQS
argo events
argo workflow
HELM
argo cd
argo rollouts
GCP
Starfish Cluster
Master
Node
Node


CI/CD
Batch Workflow
Model Workflow
Visualizer
API
Data Source
Engineer
Container Registry
Data Source
Airflow Cluster
Models/Data
S3/SQS
argo events
argo workflow
HELM
Models
jupyter
GCP
Starfish Cluster
Master
Node
Node
argo cd
argo rollouts


Streams
CI/CD
Batch Workflow
Model Workflow
Visualizer
API
Data Source
Engineer
Container Registry
Data Source
Airflow Cluster
Models/Data
S3/SQS
argo events
argo workflow
HELM
GCP
Starfish Cluster
Master
Node
Node
Models
LOG
APACHE kafka
beam
Flink
jupyter
Data Source
argo cd
argo rollouts

Thank You