Fluid
Elastic Dataset for Kubernetes

Agenda

- Why Fluid
- What's Fluid
- Fluid Architecture
- Roadmap

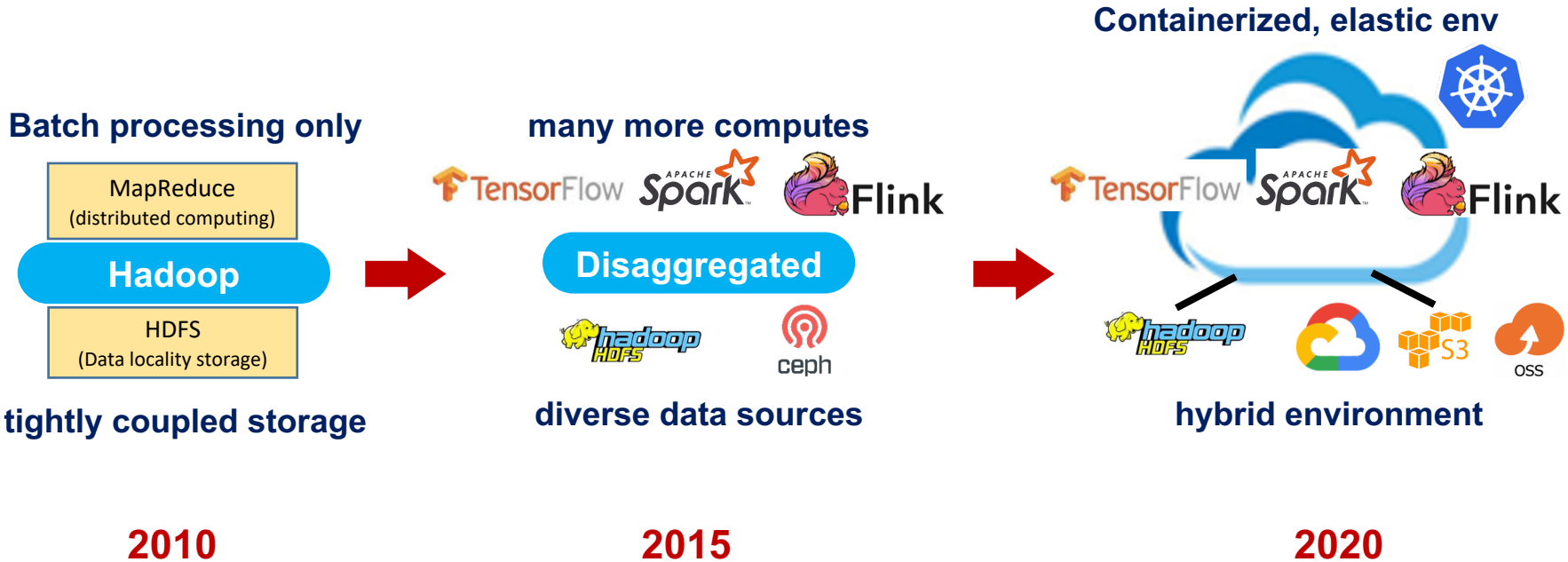
Data-intensive Applications Are Trending on Kubernetes

“By 2023, 70% of AI workloads will use Application container technology like Kubernetes” predicted by Gartner [1]

“Google Replaces YARN with Kubernetes to Schedule Apache Spark” at ApacheCon North America[2]

[1] https://www.gartner.com/en/conferences/emea/data-analytics-switzerland/featured-topics/topic-ai-machine-learning
[2] https://thenewstack.io/big-data-google-replaces-yarn-with-kubernetes-to-schedule-apache-spark/

Journey of A Data-driven Company



Technical Challenges Blocking Faster Adoption

Heterogeneous Data Source

Complexity

Disaggregated Compute & Storage

I/O Bottleneck

Locality-ignoring Scheduling

Computing Resource
(MR slots,Spark executors)

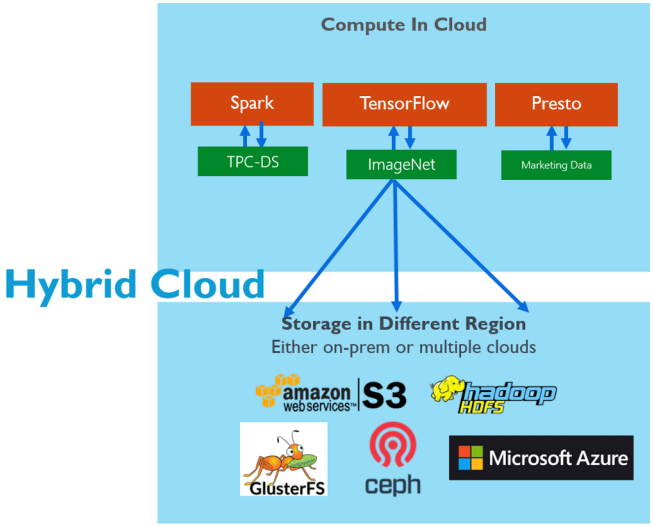
Memory Resource
(cache unit, data buffer)

Static Resource Allocation

Inefficiency

1st Challenge: Complicated Data Storage

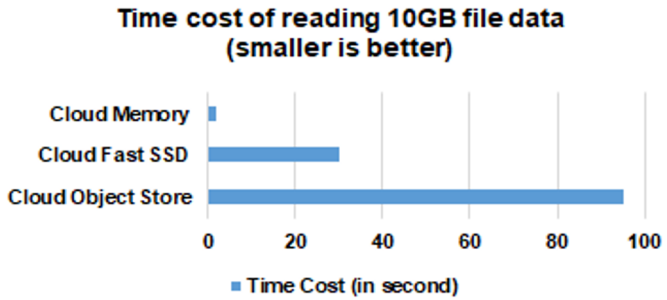
- **Heterogeneous storage**: HDFS, Ceph, S3 ...
- **Various data semantics**: files, objects, …
- **Low-level data access APIs**: DFS, S3, POSIX, and complicated settings.



Fluid introduces "`DataSet`", a high-level abstraction to applications, hiding details of heterogeneous data sources

2nd Challenge: I/O Bottleneck

- **Disaggregated compute and storage:** more I/O overhead
- **Elastic infrastructure** leads to more **remote data access**
- Concurrent data applications **competing for limited data access bandwidth**



Fluid accelerates data access with distributed cache runtimes

3rd Challenge: Locality-ignoring Scheduling

- **Workload has no aware on scheduler:**

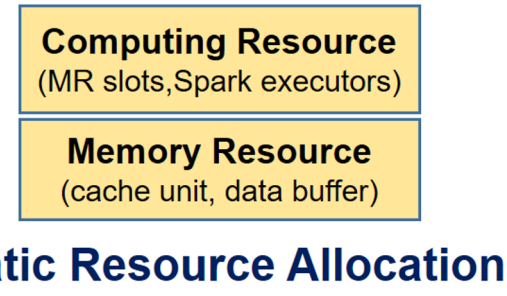
  move data or move compute?

- **No intelligent data warmup:**

  ETL is costly & error-prone

- **No data/app affinity-aware scheduling**

  for apps like Spark, Tensorflow



Fluid builds data-aware scheduling strategies on Kubernetes to coordinate applications with cache runtime automatically

What is Fluid, indeed

An Elastic Data Abstraction and Acceleration Platform in Cloud Native Environment.

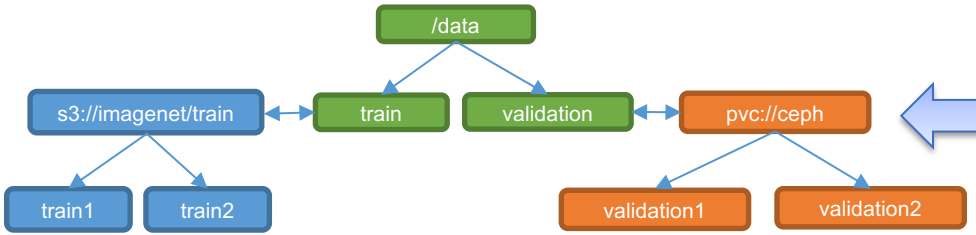
Data Abstraction
for heterogeneous data sources

Data Acceleration
with autoscaling and portable cache runtimes

Data-aware Scheduling
to improve data affinity for apps intelligently

A Quick Start on Fluid: (1) Create a “Dataset”

Edit “dataset.yaml” and run “kubectl apply -f dataset.yaml”




```yaml
apiVersion: data.fluid.io/v1alpha1
kind: Dataset
metadata:
  name: imagenet
spec:
  mounts:
  - mountPoint: s3://imagenet/train
    name: train
  - mountPoint: pvc://ceph
    name: validation
  nodeAffinity:
      required:
        - nodeSelectorTerms:
            - matchExpressions:
                -key: GPU
                 operator: In
                 values:
                 - "true"
```

s3

ceph

In GPU nodes

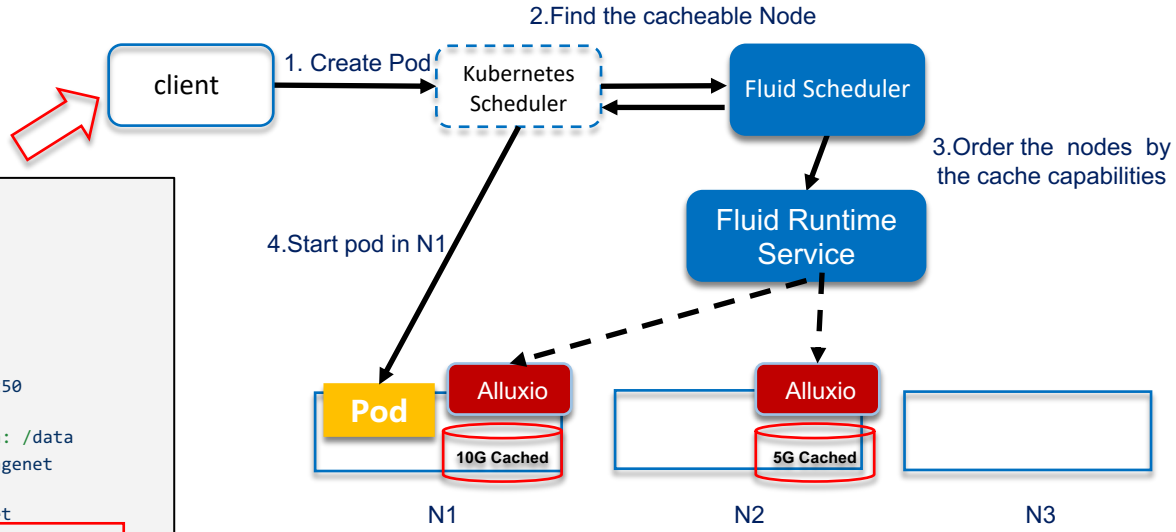
```
[root@iZuf6afslw5j6hauyzkw21Z dataset]# kubectl get dataset
NAME       UFS TOTAL SIZE   CACHED   CACHE CAPACITY   CACHED PERCENTAGE   PHASE   AGE
imagenet   183.08GiB        0.00B    100.00GiB        0.0%                Bound   23m
```

A Quick Start on Fluid: (2) Run Apps with Data Affinity

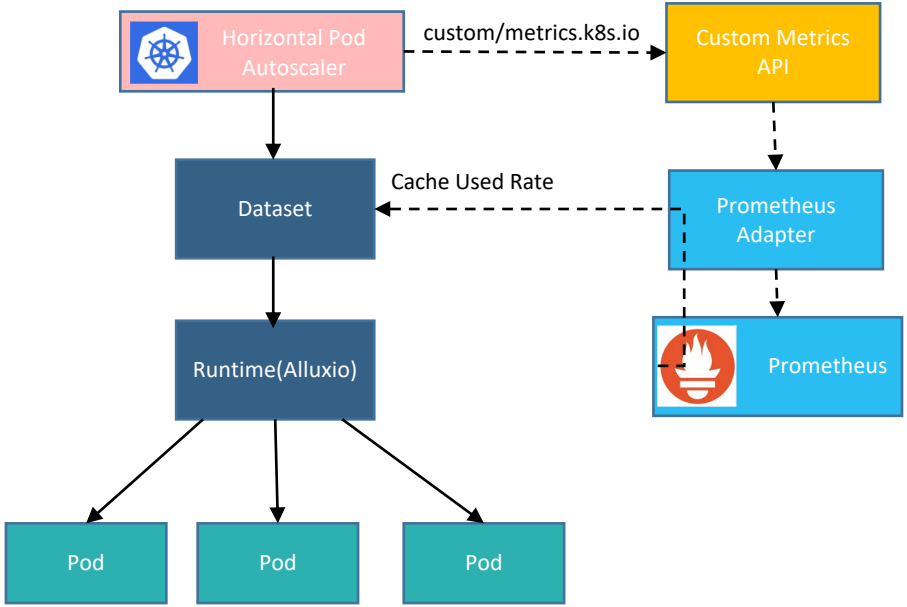
Create "pod.yaml" and run "kubectl create -f pod.yaml"

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: resnet50
spec:
   containers:
        - name: train
          image: resnet50
          volumeMounts:
            - mountPath: /data
              name: imagenet
      volumes:
        - name: imagenet
          persistentVolumeClaim:
            claimName: imagenet
```

Dataset Name

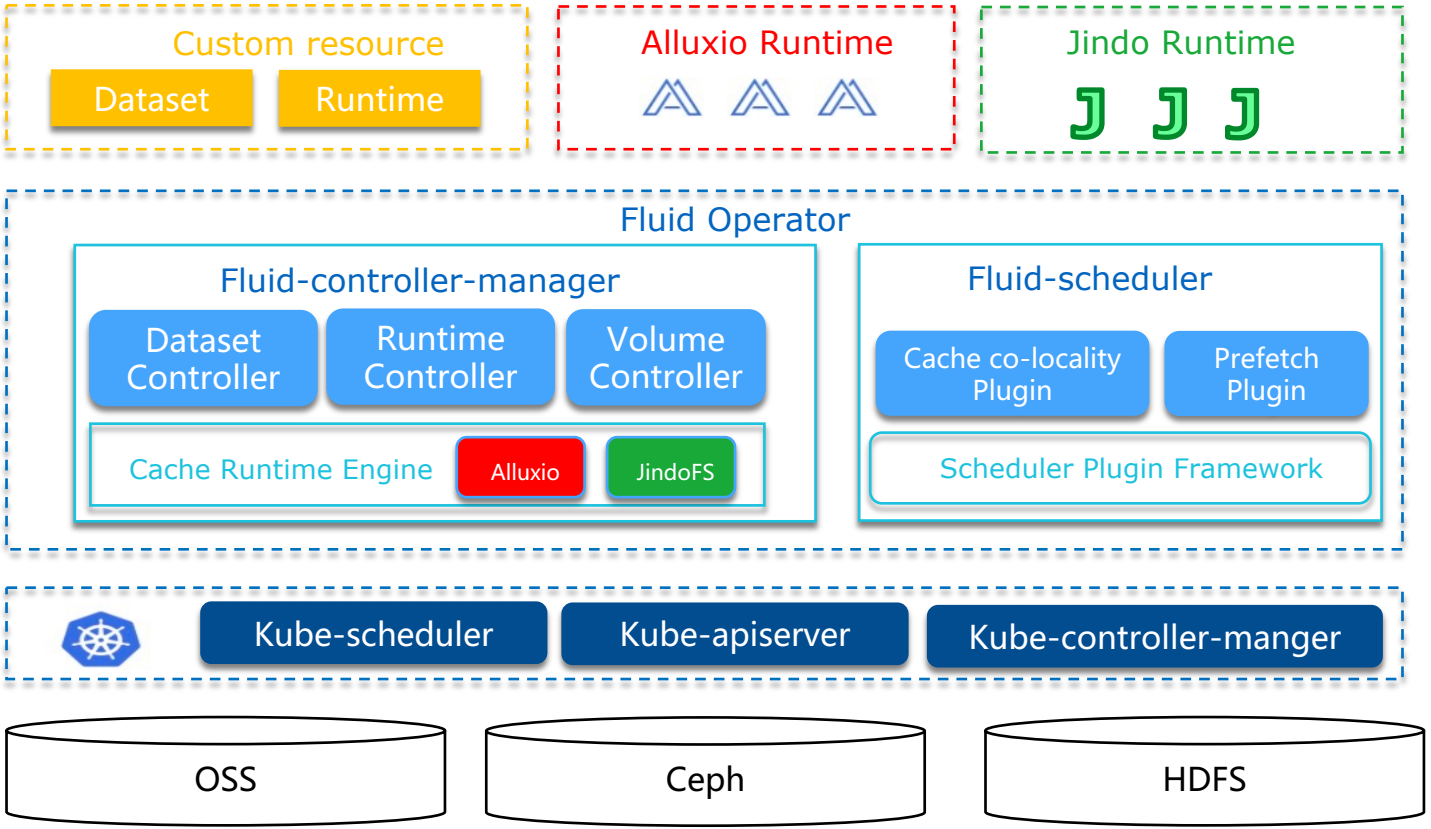


Quick Start on Fluid: Auto-scale Dataset

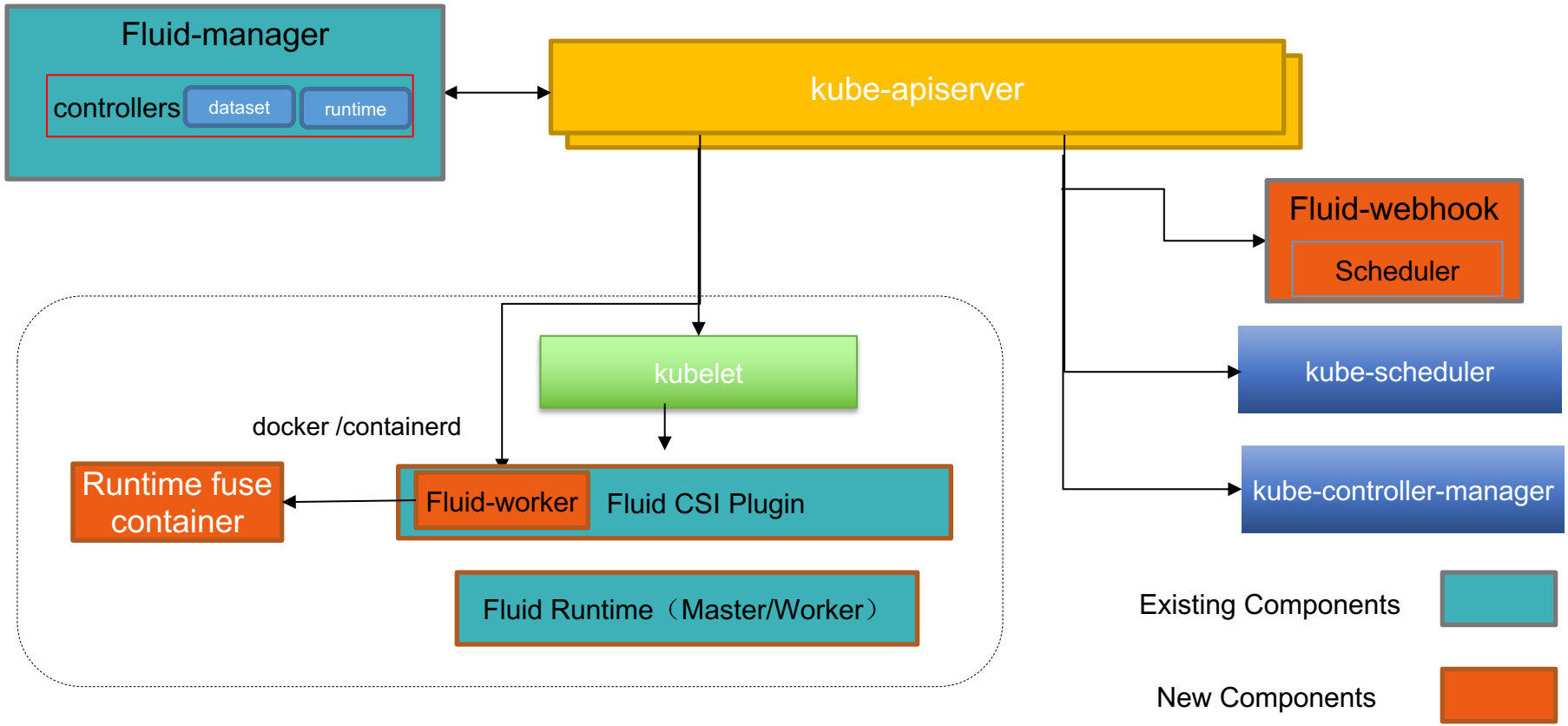



```yaml
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: imagenet-hpa
spec:
  scaleTargetRef:
    apiVersion: data.fluid.io/v1alpha1
    kind: AlluxioRuntime
    name: imagenet
  minReplicas: 2
  maxReplicas: 10
  metrics:
    - type: Object
      object:
        metric:
          name: capacity_used_rate
        describedObject:
                  apiVersion: data.fluid.io/v1alpha1
          kind: Dataset
          name: imagenet
        target:
          type: Value
          value: ”75"
```

Fluid Architecture



Fluid Architecture(New)



Existing Use Cases

- **Deep learning model training on K8s with high computing bandwidth**
  - The same training dataset is used repeatedly
  - Accessing training data is the bottleneck
- **Unified data processing pipeline with analytic engines and AI systems**
  - Saving data to external storage leads to huge I/O cost
  - Applying cross-task optimization (pipelining) on tasks is challenging
- **Elastic distributed deep learning (e.g. computing resources demand varies)**
  - Data cache capacity requirement varies during app run time

Key Scenarios

- Manage and operate datasets in large scale K8s Cluster to improve the performance of accessing of data
  - Long time Dataset for regular usage
  - Multiple datasets co-exist

- Feedbacks
  - Scheduling for large-scale cluster
  - Self-maintained
  - Big data
    - Presto/Spark

Roadmap

Fluid 0.5

- Accelerate DL on Kubernetes in a simple way
- Scale out/in Dataset
- A small set of datasets

Use case

As a data infrastructure engineer, I want to be able to setup dataset cache for AI immediately that can be easily and efficiently accessed.

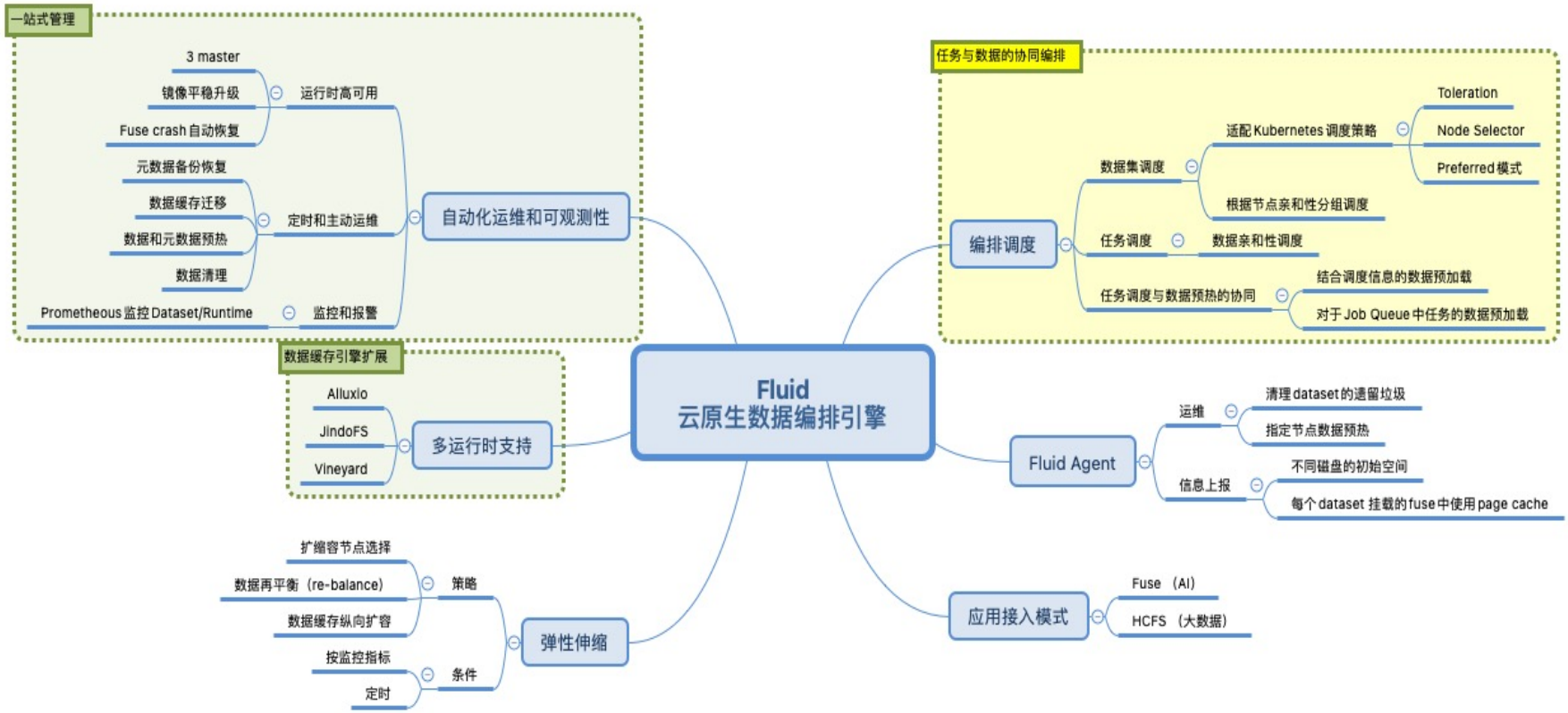
Fluid 1.0

- Enable big data on Kubernetes
- Comprehensive schedule capability for data and workload
- Dataset lifecycle management in large scale of Kubernetes

Use case

As a data infrastructure engineer, I want to run both AI and Big Data in the same K8s with data acceleration. And I don't want to take care of the data management.

Plan


Fluid
云原生数据编排引擎
一站式管理
自动化运维和可观测性
运行时高可用
3 master
镜像平稳升级
Fuse crash 自动恢复
定时和主动运维
元数据备份恢复
数据缓存迁移
数据和元数据预热
数据清理
监控和报警
Prometheous 监控 Dataset/Runtime
数据缓存引擎扩展
多运行时支持
Alluxio
JindoFS
Vineyard
弹性伸缩
策略
扩缩容节点选择
数据再平衡（re-balance）
数据缓存纵向扩容
条件
按监控指标
定时
任务与数据的协同编排
编排调度
数据集调度
适配 Kubernetes 调度策略
Toleration
Node Selector
Preferred 模式
根据节点亲和性分组调度
任务调度
数据亲和性调度
任务调度与数据预热的协同
结合调度信息的数据预加载
对于 Job Queue 中任务的数据预加载
Fluid Agent
运维
清理 dataset 的遗留垃圾
指定节点数据预热
信息上报
不同磁盘的初始空间
每个 dataset 挂载的 fuse 中使用 page cache
应用接入模式
Fuse （AI）
HCFS （大数据）