



GEdge(Griffin-Edge) Platform

- 초저지연 지능형 클라우드 엣지 SW 플랫폼 -

# 클라우드 엣지 협업 기반 ML 학습-추론 실행 관리 기술

2023.12.07

GS-Optops 커뮤니티 멤버

조 정 현(junghyuncho@sk.com)

“GEdge Platform”은 클라우드 중심의 엣지 컴퓨팅 플랫폼을 제공하기 위한  
핵심 SW 기술 개발 커뮤니티 및 개발 결과물의 코드명입니다.

- New Leap Forward of  
GEdge Platform Community 7<sup>th</sup> Conference (GEdge Platform v4.0 Release) -

# Contents

---

- I 클라우드 엣지 협업 기반 ML학습-추론 실행
- II 클라우드 엣지 기반 ML개발 환경 자동 구축
- III 워크플로우 기반 학습-추론 실행 환경 지원
- IV 클라우드 엣지 기반 학습-추론 실행 결과

# GEdge 플랫폼 내 gs-optops의 역할

## 초저지연 지능형 클라우드 엣지 플랫폼 (GEdge Platform)

### 클라우드 엣지 관리 플랫폼 (GM : GEdge Management Platform)

플랫폼 관리 도구 프레임워크 (GM-Tool)

Framework I/F

플랫폼 관리 기능 프레임워크 (GM-Center)

Platform I/F

### 지능형 서비스 운용 프레임워크 (GS-AI)

엣지 AI 서비스 환경  
(GS-AIflow)

엣지 협업 학습 환경  
(GS-Optops)

Framework I/F

### 서비스 협업 프레임워크 (GS-Link)

협업 게이트웨이  
(GS-Linkgw)

협업 정책 생성  
(GS-Linkhq)

Framework I/F

### 초저지연 데이터 처리 프레임워크 (GS-Engine)

엣지 전용 스케줄러  
(GS-Scheduler)

엣지 메시지 브로커  
(GS-Broker)

### 클라우드 엣지 서비스 플랫폼 (GS : GEdge Service Platform)



**클라우드 엣지 협업 기반 ML학습-추론 실행**



## 기존 기술 문제점

제한적 가속기 지원중앙 집중형 학습  
→ 지능형 엣지 환경 부적합

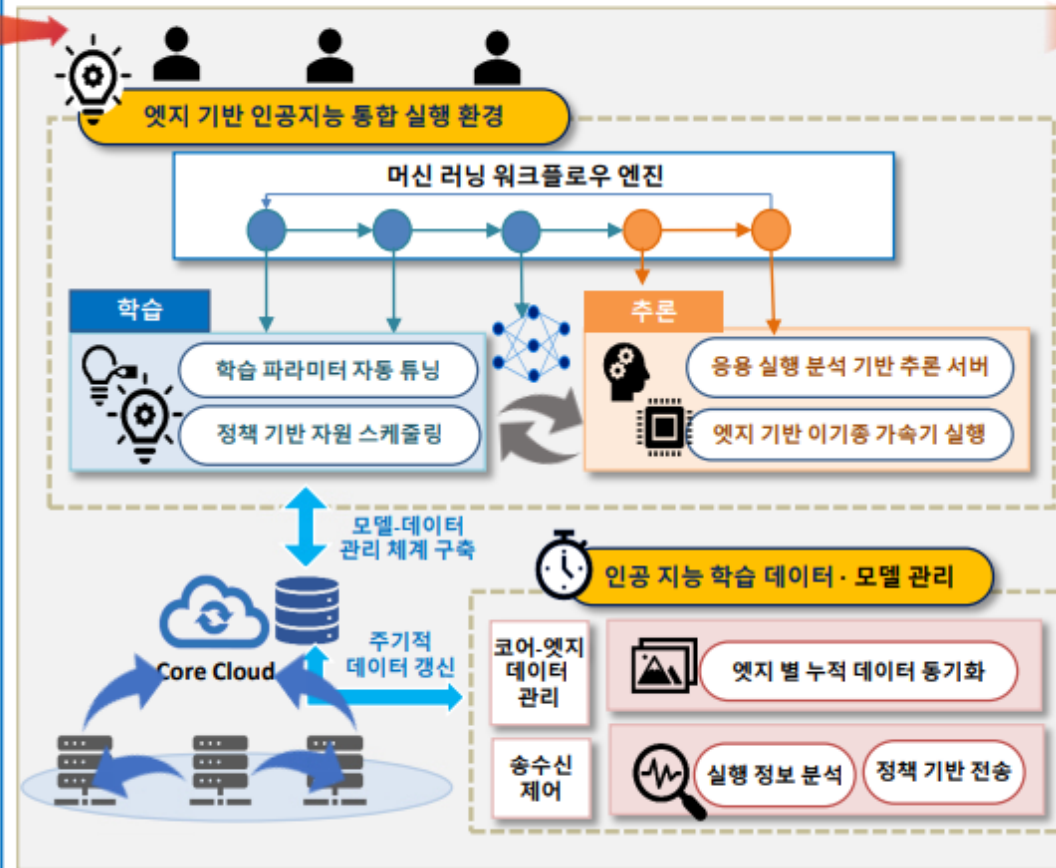


- 학습-추론 통합 실행 관리 솔루션 한계 (오픈소스-일부 고도화 기능 미공개)
- 이종 가속기 지원 추론 클라우드 부재



- 중앙 집중형 학습 데이터 저장 관리 방식
- 학습 다양성에 비적응적 데이터 처리

## 핵심 개발 내용



## 개선 기술 특징점

최적 학습·고속 추론 지원  
엣지 맞춤형 AI 플랫폼 제공

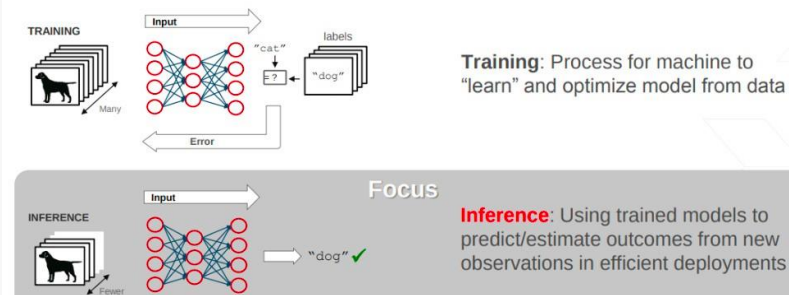


- 엔터프라이즈급 인공지능 통합 실행 환경 기반 학습 처리 속도 향상  
: 용이한 재학습 환경으로 정확도 향상 기여  
: 엣지 기반 이기종 가속기 실행 기능 제공

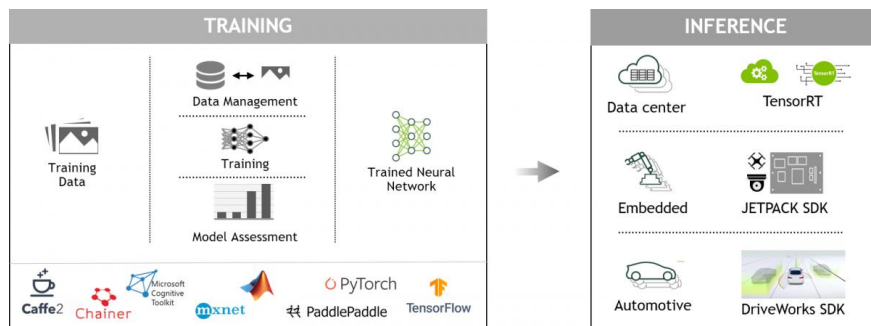


- 협업 기반 인공지능 학습 데이터·모델 관리  
: 모델-데이터 저장 체계구축으로 확장성 지원  
: 학습 기법별 데이터 전송 정책 관리  
: 엣지 학습 누적 데이터 동기화 관리

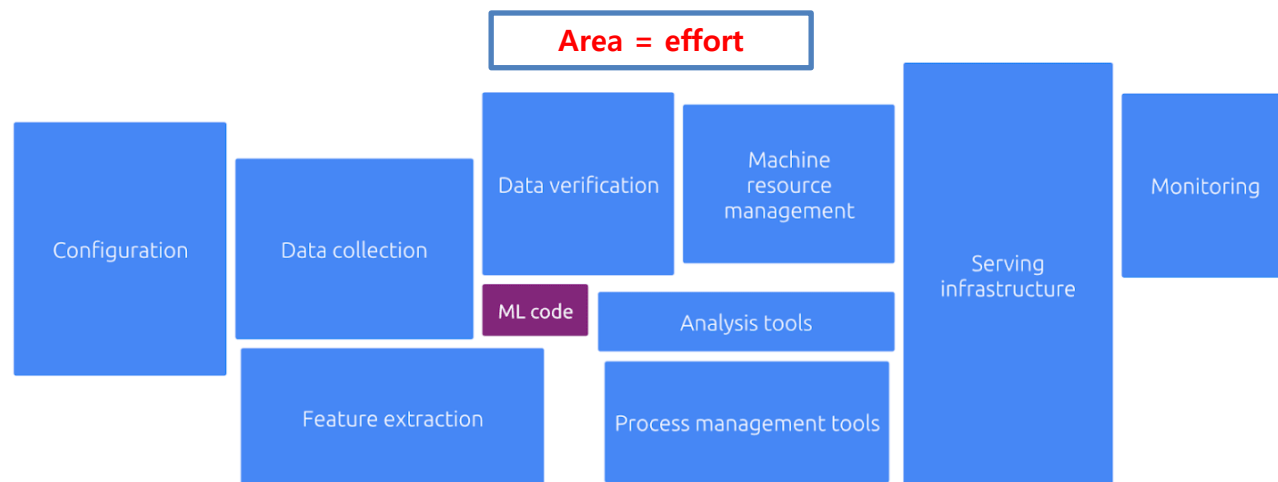
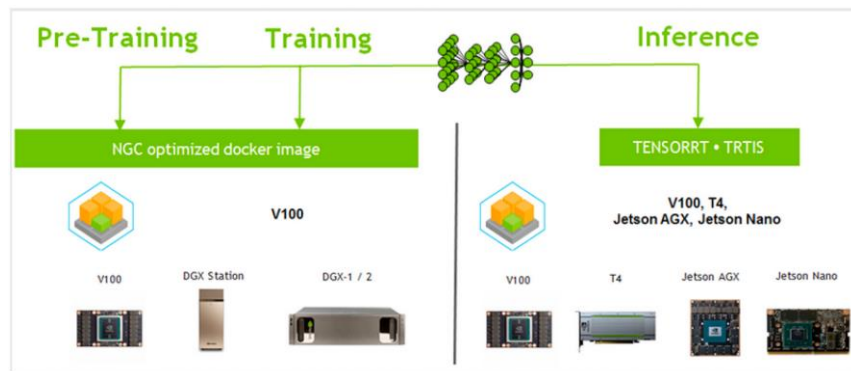
- 학습 (with Large-Scale Dataset)
  - Weights Updates for Cost Function Minimization
- 추론 (Real-Word Execution)
  - Inference Results based on Updated Weights in Neural Networks



<https://www.xilinx.com/applications/ai-inference/difference-between-deep-learning-training-and-inference.html>

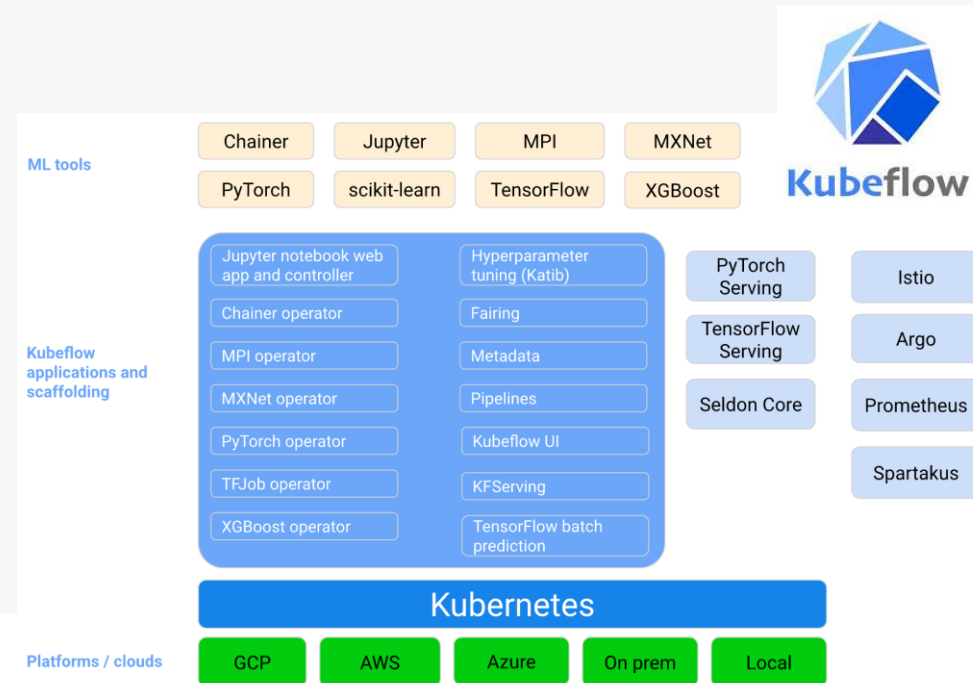


<https://www.edge-ai-vision.com/2019/10/automatic-defect-inspection-using-the-nvidia-end-to-end-deep-learning-platform/>



(<https://ubuntu.com/blog/kubernetes-for-data-science-meet-kubeflow>)

- 사용자 요청(On-demand)에 따라 편리하게 생성되는 ML학습-추론 실행 환경 자동 생성
  - 사용자가 원하는 Resource 요구사항에 따라 ML학습-추론 실행 환경 구성
  - GEDGE Platform상에서 구동되기 위해 Kubernetes 호환성 지원 필요
  - 사용자 별 자원 격리(Resource Isolation)지원 필요
- Workflow 기반의 ML학습-추론 실행 환경 구성
  - ML Workflow / MLOps 파이프라인 등, 다양한 Platform과 Tool 출시되고 있음
- Kubeflow (<https://www.kubeflow.org/>)
  - 엔드투엔드(End-to-End) AI 플랫폼
  - 머신러닝 워크플로우의 머신러닝 모델 학습부터 배포 단계까지
  - 모든 작업에 필요한 도구와 환경을 쿠버네티스(Kubernetes) 위에서 쿠브플로우 컴포넌트로 제공
    - composability: 머신러닝 워크플로우의 컴포넌트들을 프로젝트의 요구 조건(버전, 라이브러리 등)에 따라 구성할 수 있음
    - portability: 로컬, 온프레미스, 클라우드 등 다양한 환경에 배포 가능
    - scalability: 쿠버네티스의 기능을 활용하여 CPU, GPU, TPU 등의 자원 확장/축소 가능



Kubeflow Conceptual Overview :  
 (<https://www.kubeflow.org/docs/started/architecture/#conceptual-overview>)



# 클라우드 엣지 협업 기반 ML 학습-추론 실행 관리 기술

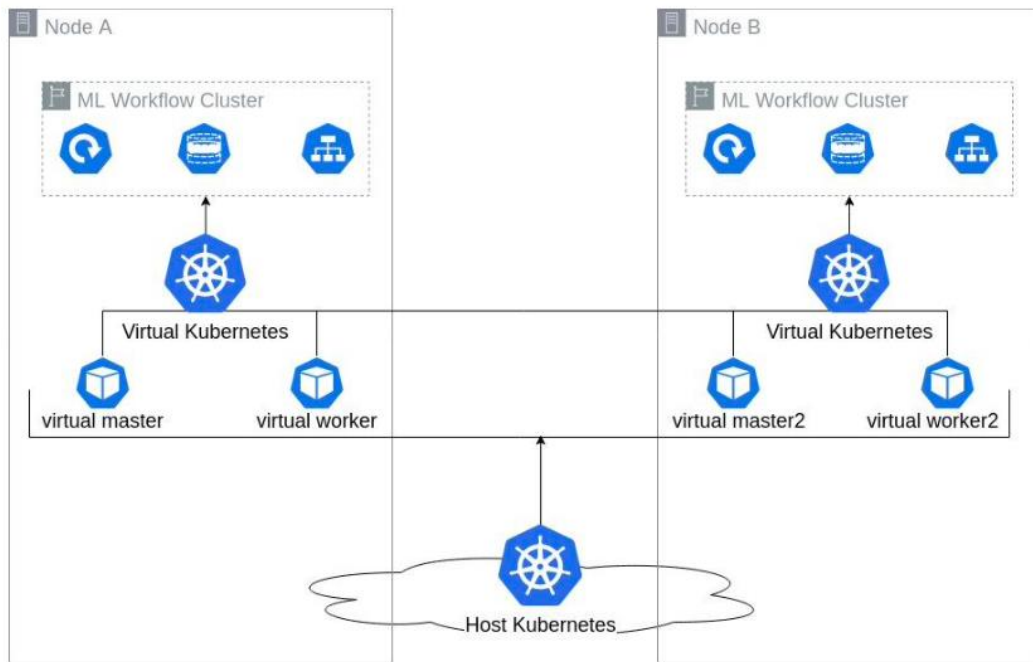




# II 클라우드 엣지 기반 ML개발 환경 자동 구축

- GS-Optops Virtual Kubernetes 개발

- Host 서버에 설치된 Kubernetes 버전에 의존성 없이, On-demand로 Kubernetes 및 Resource 환경 자동 구성 지원
  - 사용자로부터 자원 요구사항, 설치 버전 요구사항 입력 받음
- Kubeflow 를 포함한 형상으로 배포 지원
  - Kubeflow Compatibility Matrix 반영



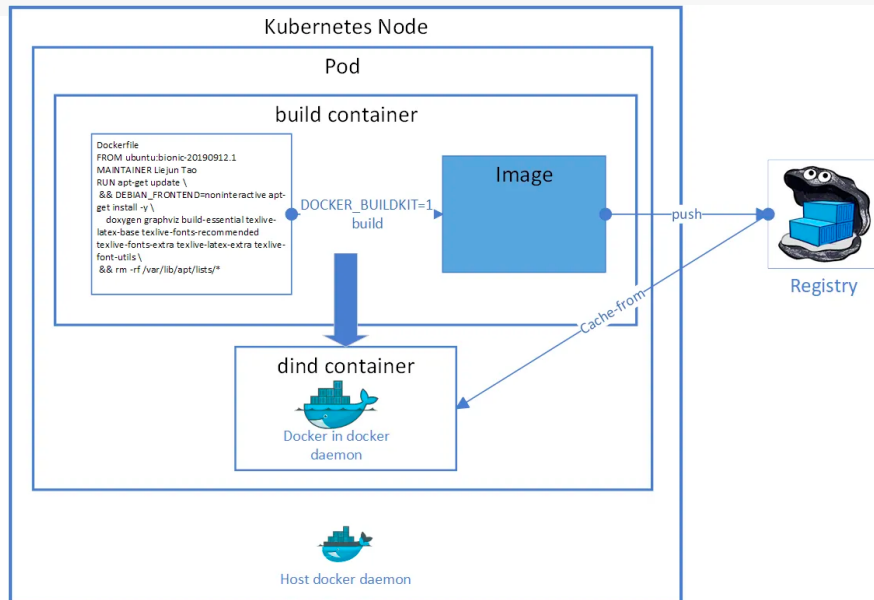
GS-Optops Virtual Kubernetes

Kubernetes Versions	Kubeflow 0.4	Kubeflow 0.5	Kubeflow 0.6	Kubeflow 0.7
1.11	compatible	compatible	incompatible	incompatible
1.12	compatible	compatible	incompatible	incompatible
1.13	compatible	compatible	incompatible	incompatible
1.14	compatible	compatible	compatible	compatible
1.15	incompatible	compatible	compatible	compatible
1.16	incompatible	incompatible	incompatible	incompatible

<https://v0-7.kubeflow.org/docs/started/k8s/overview/>

## • vk8s 개발을 위한 핵심 요소 기술

- vNode 개념 적용 : Kubernetes Pod 안에 또 다른 kubernetes 띄우는 형태
- DinD(DockerinDocker) : 도커 컨테이너 내에서 도커 데몬을 추가로 동작 방식으로 컨테이너가 컨테이너를 컨트롤할 수 있음
- k8s Custom Resource 신규 정의
  - 쿠버네티스에서는 오브젝트 종류를 직접 정의해 사용할 수 있음. 즉, API를 직접 확장해 사용할 수 있도록 인터페이스를 제공
  - CRD(Custom Resource Definition)와 Operator를 이용한 커스텀 Custom Resource 생성 적용



Fast Docker build in Kubernetes  
(<https://medium.com/swlh/fast-docker-build-in-kubernetes-f52088854f45>)

```

apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.6.1
  creationTimestamp: null
  name: vk8s.vk8s.skt.co.kr
spec:
  group: vk8s.skt.co.kr
  names:
    kind: Vk8s
    listKind: Vk8sList
    plural: vk8s
    singular: vk8s
  scope: Namespaced
  versions:
  - name: v1alpha1
    schema:
      openAPIV3Schema:
        description: Vk8s is the Schema for the vk8s API
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this representation of
            type: string
            kind:
            description: 'Kind is a string value representing the REST resource this object
            type: string
            metadata:
              type: object
          spec:
            description: Vk8sSpec defines the desired state of Vk8s
            properties:
              accessPodImage:
                description: AccessPodImage is image of access pod for vk8s cluster
                type: string
              accessPodPort:
                description: AccessPodPort is ssh export port number of access pod
                format: int32
                type: integer
              kubeflow:
                description: Kubeflow is kubeflow's version to install
                properties:

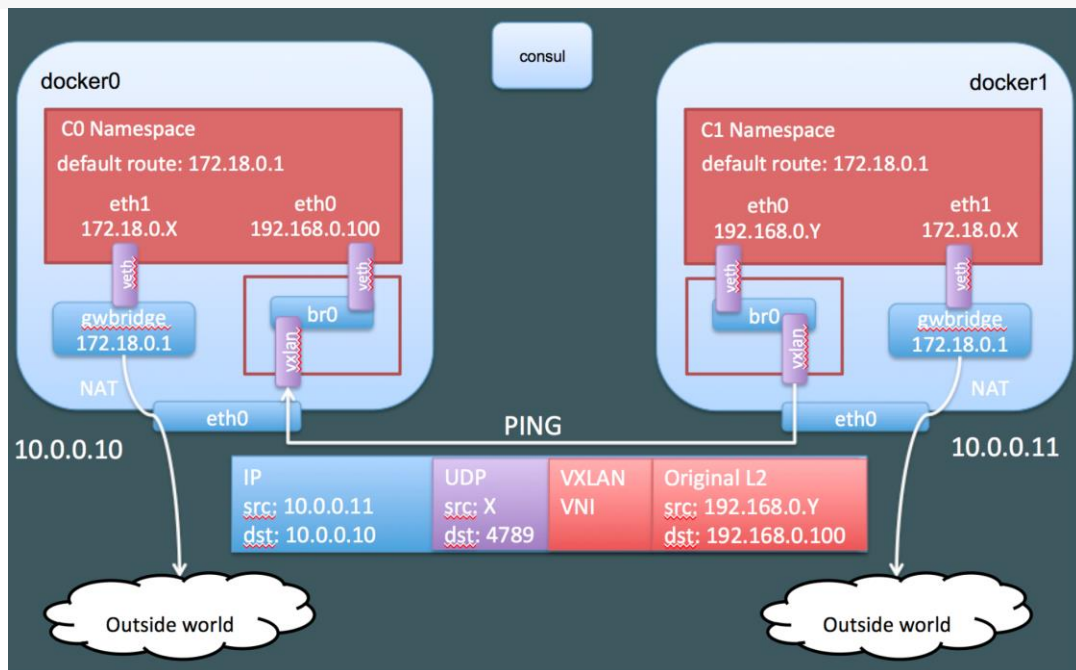
```

GS-Optops vk8sd의 위한 Custom Resource Definition



# 워크플로우 기반 학습-추론 실행 환경 지원

- Kubeflow in vk8s 개발을 위한 핵심 요소 기술
  - Kubeflow Compatibility Matrix 반영 : 사용자가 Kubeflow 버전 선택 시, 자동 매핑
  - Kubeflow 서비스 접속을 위한 Network Issue 해결
    - docker overlay network를 사용하는 컨테이너 네트워크 인터페이스는 eth0, eth1 2개가 있음
    - eth1은 외부와의 통신을 위한 네트워크 장치로 컨테이너간 통신은 불가능
    - eth0은 overlay network이며, 컨테이너간 통신에 사용
    - Kubeflow in vk8s을 위해 CNI인 calico에 eth1가 아닌 eth0을 사용해야 한다고 명시



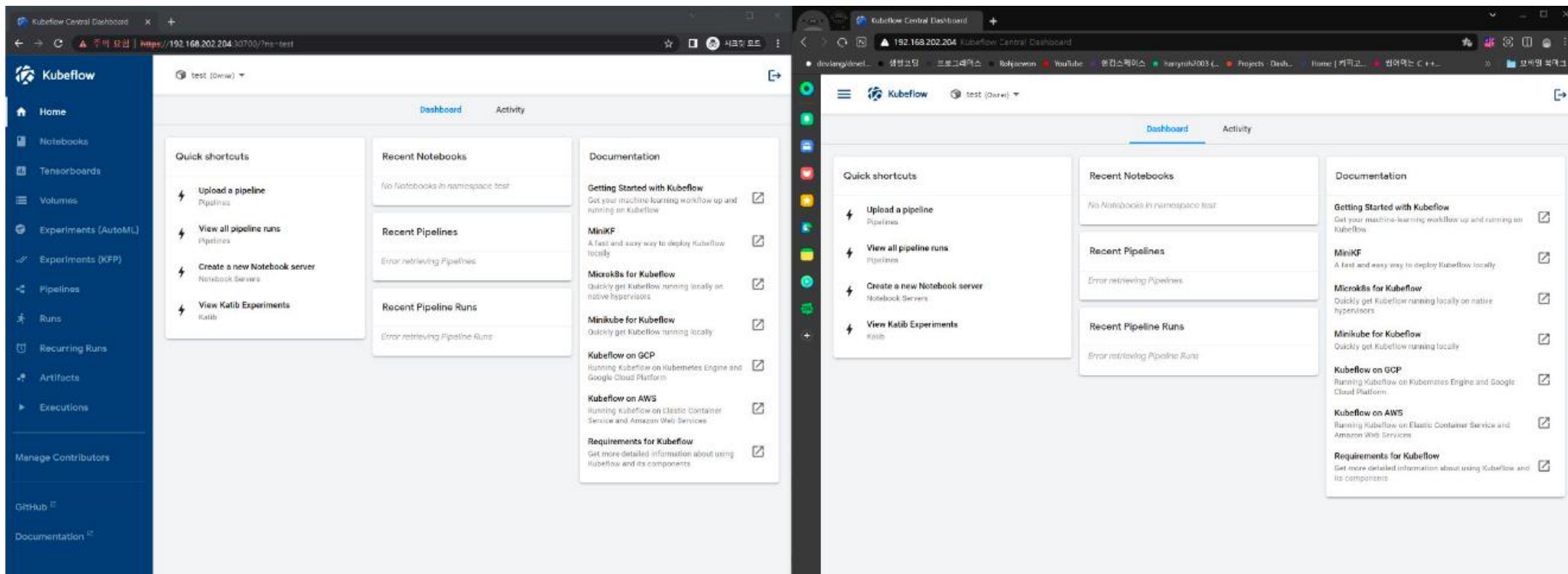
```
kind: DaemonSet
...
spec:
  template:
    spec:
      containers:
        - name: calico-node
          env:
            - name: IP_AUTODETECTION_METHOD
              value: "interface=eth0"
```

- vk8s 기반의 KubeFlow 실행환경 자동 배포 기술
  - 사용자로부터 리소스, kubeflow 버전 등의 정보를 yaml파일로 받도록 유연하게 구성
  - 입력받은 요구사항은 yaml화 되며, 동적으로 생성

```
(base) → vk8s git:(master) x cat config/samples/sample_3.yaml
apiVersion: vk8s.skt.co.kr/v1alpha1
kind: Vk8s
metadata:
  name: sample3
spec:
  nodes:
  - name: sample-master3
    image: harbor.dudaji.com/vk8s/kink:v1.20
    role: master
    resources:
      requests:
        cpu: "1"
        memory: "16Gi"
      limits:
        cpu: "10"
        memory: "40Gi"
    kubernetes:
      podNetworkCidr: 80.244.0.0/16
      serviceCidr: 80.96.0.0/12
    kubeflow:
      version: v1.3.0
      email: test@example.com
      password: test!!22
  ports:
  - name: kubeflow
    port: 30000
    targetPort: 30000
(base) → vk8s git:(master) x kubectl apply -f config/samples/sample_3.yaml
vk8s.vk8s.skt.co.kr/sample3 created
```

```
(base) → vk8s git:(master) x cat config/samples/sample_4.yaml
apiVersion: vk8s.skt.co.kr/v1alpha1
kind: Vk8s
metadata:
  name: sample4
spec:
  nodes:
  - name: sample-master4
    image: harbor.dudaji.com/vk8s/kink:v1.20
    role: master
    resources:
      requests:
        cpu: "1"
        memory: "16Gi"
      limits:
        cpu: "10"
        memory: "40Gi"
    kubeflow:
      version: v1.4.0
      email: test@example.com
      password: test!!22
  ports:
  - name: kubeflow
    port: 30000
    targetPort: 30000
(base) → vk8s git:(master) x kubectl apply -f config/samples/sample_4.yaml
vk8s.vk8s.skt.co.kr/sample4 created
```

- vk8s 기반의 Kubeflow 실행환경에서 사용자 별 작업 수행
  - (관리자) vk8s의 설치 단계 상태, 설치 히스토리 모니터링가능
  - (사용자) 사용자 별 서비스 포트(NodePort)를 통해 Kubeflow 접속 및 작업 실행 가능



(좌) 사용자 A : kubeflow v1.3 실행

(우) 사용자 B : kubeflow v1.4 실행

# IV 클라우드 엣지 기반 학습-추론 실행 결과



- 다양한 Kubernetes 버전에서 vK8S 실행 호환성 검증 완료
  - Host 서버에 설치된 Kubernetes v1.23.8에서
  - 학습 실행 환경(Ubuntu)기반의 vk8s-1-19-ubuntu를 vNode로 실행

Nodes(all) [7]			
NAME↑	STATUS	ROLE	VERSION
ddp-scale-gpu01	Ready	worker	v1.23.8
ddp-scale-gpu02	Ready	worker	v1.23.8
ddp-scale-gpu03	Ready	worker	v1.23.8
ddp-scale-gpu04	Ready	worker	v1.23.8
ddp-scale-master01	Ready	control-plane,master	v1.23.8
ddp-scale-master02	Ready	control-plane,master	v1.23.8
ddp-scale-master03	Ready	control-plane,master	v1.23.8

Vk8s(all) [2]	
NAMESPACE↑	NAME
default	vk8s-1-19-ubuntu
default	vk8s-1-20-ubuntu

Pods(all) [5] </vk8s>					
NAMESPACE↑	NAME	DE	READY	RESTARTS	STATUS
default	vk8s-1-19-ubuntu-node-0-0	●	1/1	0	Running
default	vk8s-1-20-ubuntu-node-0-0	●	1/1	0	Running
vk8s-system	ip-manager-678f54c664-psds5	●	1/1	3	Running
vk8s-system	mysql-0	●	1/1	0	Running
vk8s-system	vk8s-controller-manager-7cc98744d4-tb95j	●	2/2	1	Running



- vK8S 실행 안정성 테스트 완료

- vk8s 업데이트 테스트

- vk8s 리소스의 레이블을 추가하고, 추가한 레이블이 StatefulSet에도 똑같은 레이블로 생기는지 확인

#### Vk8s 업데이트 테스트

Vk8s 리소스의 레이블을 새로 추가하고 추가한 레이블이 StatefulSet에도 똑같은 레이블이 생기는지 확인함으로써 업데이트 여부를 테스트한다.

##### ▶ 테스트 코드

```
apiVersion: vk8s.sktelecom.com/v1alpha1
kind: Vk8s
metadata:
  annotations:
    cluster-network-connection-status: "true"
    kubectrl.kubernetes.io/last-applied-configuration:
      {"apiVersion":"vk8s.sktelecom.com/v1alpha1","kind":"Vk8s","metadata":{"name":"test!!22","version":"v1.3.0"},"spec":{"nodes":[{"image":"1Gi"}],"role":"master","tolerations":[{"effect":"NoSchedule"}]}}
    creationTimestamp: "2023-10-27T15:39:37Z"
    generation: 1
  labels:
    update: test
  name: vk8s-1-19-ubuntu
  namespace: default
  resourceVersion: "305046"
  uid: 090c5044-0aca-40e4-8087-052eda21f5df
```

vk8s 리소스(왼쪽 사진)에서 새로운 라벨을 추가하면 관련된 statefulset 리소스(오른쪽 사진)에 자동으로 라벨이 업데이트 되는 것을 확인한다.

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  annotations:
    cluster-network-connection-status: "true"
    kubectrl.kubernetes.io/last-applied-configuration:
      {"apiVersion":"vk8s.sktelecom.com/v1alpha1","kind":"Vk8s","metadata":{"name":"test!!22","version":"v1.3.0"},"spec":{"nodes":[{"image":"1Gi"}],"role":"master","tolerations":[{"effect":"NoSchedule"}]}}
    creationTimestamp: "2023-10-27T15:39:37Z"
  finalizers:
    - foregroundDeletion
  generation: 1
  labels:
    app: vk8s
    clusterName: vk8s-1-19-ubuntu
    role: master
    sidecar.istio.io/inject: "false"
    update: test
  name: vk8s-1-19-ubuntu-node-0
```

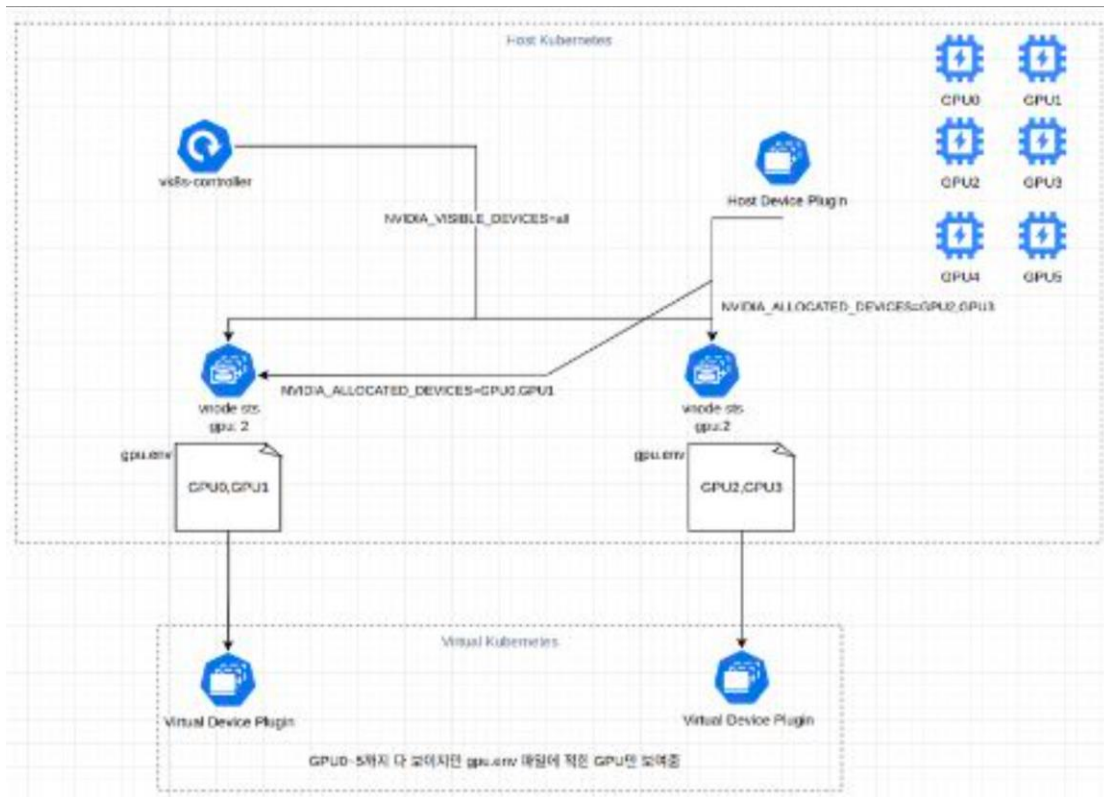
```
STEP: Validating that vk8s's label updated successfully. @ 10/20/23 00:36:10.611
Validating that vk8s's label updated successfully
running dir: /home/dudaj/nopro-workspace/gitRepo/vk8s/vk8s
running: kubectrl label vk8s vk8s-1-19-ubuntu update=test --n vk8s-system
running dir: /home/dudaj/nopro-workspace/gitRepo/vk8s/vk8s
running: kubectrl get vk8s vk8s-1-19-ubuntu -o jsonpath='{range .spec.nodes[*]}{.name}{"\n"}' --n vk8s-system
stdoutNames: 1, [vk8s-1-19-ubuntu-node-0]
running dir: /home/dudaj/nopro-workspace/gitRepo/vk8s/vk8s
running: kubectrl get sts vk8s-1-19-ubuntu-node-0 -o jsonpath='{.metadata.labels}' --n vk8s-system
labels: map[app:vk8s clusterName:vk8s-1-19-ubuntu role:master sidecar.istio.io/inject:false update:test vnode:vk8s-1-19-ubuntu-node-0]vk8s's label is updated successfully!!
```

```
status:
  conditions:
    - lastProbeTime: "2023-10-27T09:34:27Z"
      message: creating node
      phase: Waiting
    - lastProbeTime: "2023-10-27T09:34:35Z"
      message: configuring master node
      phase: Installing
    - lastProbeTime: "2023-10-27T09:35:28Z"
      message: installing kubeflow
      phase: Installing
    - lastProbeTime: "2023-10-27T09:40:35Z"
      message: Running
      phase: Running
  message: Running
  phase: Running
  vnodeKubernetesSetupStatuses:
    vk8s-1-19-ubuntu-node-0:
      isKubernetesSetup: Success
      status: Running
```

conditions에서 vk8s 설치 단계들의 history를 확인할 수 있다.

vnodeKubernetesSetupStatuses에선 vnode pod들의 상태와 쿠버네티스 설치 여부도 확인할 수 있다.

- vK8S 환경에서 GPU isolation 검증 완료
  - 사용자 요구사항 : GPU1개 요청한 경우
  - GPU가 다수 개 장착되어 있는 GPU 노드에서 Privileged Pod는 모든 GPU에 접근할 수 있지만, vk8s에서는 사용자 요청에 따라 GPU 1개만 사용하도록 제한



```
[svcapp_su@DDP-SCALE-GPU01 ~]$ nvidia-smi -L
GPU 0: Tesla P100-PCIE-16GB (UUID: GPU-ba75ada8-e606-69a6-307a-cca76bdb7f4a)
GPU 1: Tesla P100-PCIE-16GB (UUID: GPU-84eb8f4b-6a92-a11f-803f-74ac878b9c29)
```

Real(Host) Node 정보 : GPU 총 2개

```
(inner) root@vk8s-1-20-ubuntu-node-0-0:/$ nvidia-smi -L
GPU 0: Tesla P100-PCIE-16GB (UUID: GPU-ba75ada8-e606-69a6-307a-cca76bdb7f4a)
GPU 1: Tesla P100-PCIE-16GB (UUID: GPU-84eb8f4b-6a92-a11f-803f-74ac878b9c29)
```

vNode는 Real Node의 모든 GPU를 바라보고 접속 할 수 있으므로 GPU는 총 2개로 보임

```
Allocatable:
cpu: 8
ephemeral-storage: 423381173981
hugepages-1Gi: 0
hugepages-2Mi: 0
memory: 10956948Ki
nvidia.com/gpu: 1
pods: 110
```

사용자가 생성한 Pod : 사용자가 요청한 GPU 1개만 확인 됨



- vk8s환경에서 kubeflow 동작 확인
  - 특정 버전의 kubernetes와 kubeflow를 설치하고, vk8s 리소스 생성 여부 확인 완료
  - 버전 별로 vk8s안에 설치된 kubeflow의 모든 pod들의 상태가 Running인 것을 확인

```

NAME+          AGE
vk8s-1-19-ubuntu 2m37s

STEP: Creating an instance of the Vk8s(CR) @ 10/28/23 00:30:38.016
Creating an instance of the Vk8s with version 1-19
running dir: /home/dudaji/nopro-workspace/gitRepo/vk8s/vk8s
running: kubectl apply -f /home/dudaji/nopro-workspace/gitRepo/vk8s/vk8s/config/samples/vk8s-1-19.yaml -n vk8s-system
STEP: Validating that vk8s status.phase=Running. @ 10/28/23 00:30:38.094
Validating that vk8s is Running (including kubeflow)
running dir: /home/dudaji/nopro-workspace/gitRepo/vk8s/vk8s
running: kubectl get vk8s vk8s-1-19-ubuntu -o jsonpath={.status.phase} -n vk8s-system
Vk8s status: Waiting
running dir: /home/dudaji/nopro-workspace/gitRepo/vk8s/vk8s
running: kubectl get vk8s vk8s-1-19-ubuntu -o jsonpath={.status.phase} -n vk8s-system
Vk8s status: Installing
running dir: /home/dudaji/nopro-workspace/gitRepo/vk8s/vk8s
running: kubectl get vk8s vk8s-1-19-ubuntu -o jsonpath={.status.phase} -n vk8s-system
Vk8s status: Installing
running dir: /home/dudaji/nopro-workspace/gitRepo/vk8s/vk8s
running: kubectl get vk8s vk8s-1-19-ubuntu -o jsonpath={.status.phase} -n vk8s-system
Vk8s status: Installing
running dir: /home/dudaji/nopro-workspace/gitRepo/vk8s/vk8s
running: kubectl get vk8s vk8s-1-19-ubuntu -o jsonpath={.status.phase} -n vk8s-system
Vk8s status: Installing
running dir: /home/dudaji/nopro-workspace/gitRepo/vk8s/vk8s
running: kubectl get vk8s vk8s-1-19-ubuntu -o jsonpath={.status.phase} -n vk8s-system
Vk8s status: Running
STEP: Validating that kubeflow is installed successfully. @ 10/28/23 00:36:38.537
Validating that kubeflow is installed successfully
running dir: /home/dudaji/nopro-workspace/gitRepo/vk8s/vk8s
running: kubectl get svc vk8s-1-19-ubuntu -o jsonpath={.spec.ports[0].nodePort} -n vk8s-system
running dir: /home/dudaji/nopro-workspace/gitRepo/vk8s/vk8s
running: curl -k https://localhost:32704
Kubeflow is installed successfully!!

```

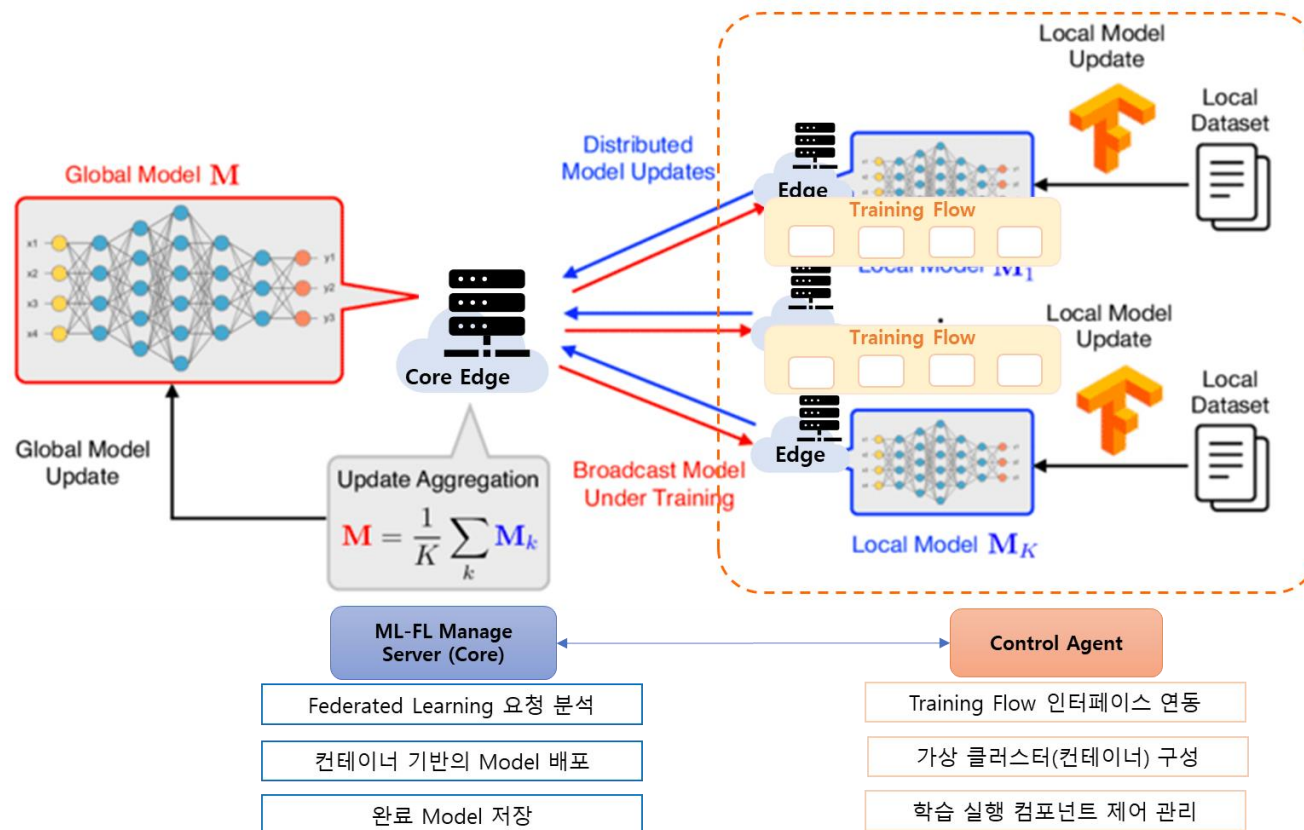
```

root@vk8s-1-19-ubuntu-node-0-0:/$ kubectl get po -A

```

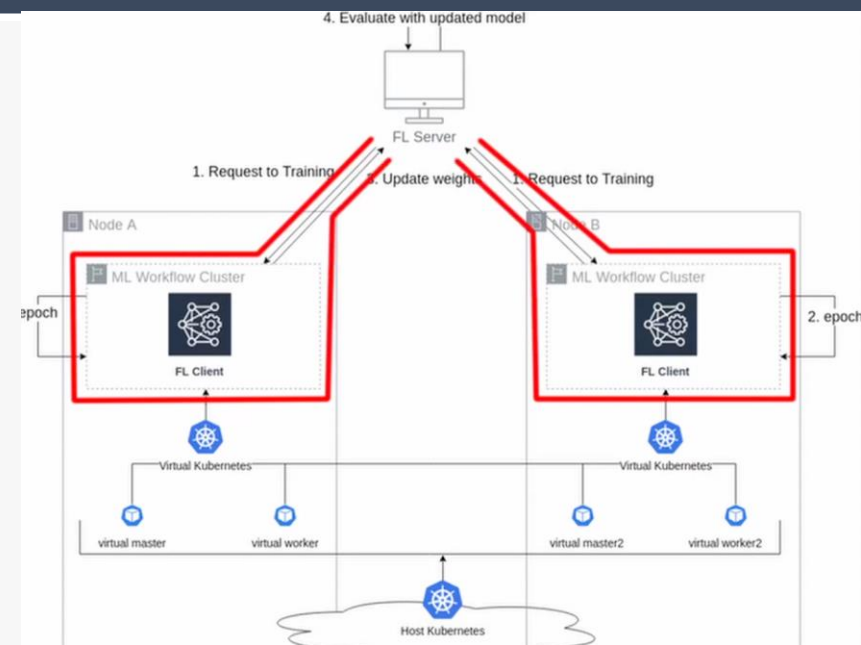
NAMESPACE	NAME	READY	STATUS	RESTARTS
auth	dex-5478bcbcf-kmbq5	1/1	Running	0
cert-manager	cert-manager-8478b94878-slhk7	1/1	Running	0
cert-manager	cert-manager-cainjector-84f647b4f-zbh9q	1/1	Running	0
cert-manager	cert-manager-webhook-58d5f7c8-d2qkp	1/1	Running	0
istio-system	authservice-0	1/1	Running	0
istio-system	cluster-local-gateway-344bd67886-6hl9p	1/1	Running	0
istio-system	istio-ingressgateway-6dcc9965c-wnnf8	1/1	Running	0
istio-system	istiod-78d77b4fd-hkwwb	1/1	Running	0
knative-eventing	broker-controller-657dd45687-bfdj8	1/1	Running	0
knative-eventing	eventing-controller-b7f75c7cf-df5z6	1/1	Running	0
knative-eventing	eventing-webhook-85ccbb8f58-l7fpv	1/1	Running	0
knative-eventing	in-c-controller-7cc6ddc88c-wcdm2	1/1	Running	0
knative-eventing	in-c-dispatcher-5c7c4d4fdc-m22js	1/1	Running	0
knative-serving	activator-86dcc59df7-5c9df	1/1	Running	0
knative-serving	autoscaler-5fc44c5996-wswpk	1/1	Running	0
knative-serving	controller-355677885d-9p2kv	1/1	Running	0
knative-serving	istio-webhook-799ff6b44f-ggbf4	1/1	Running	0
knative-serving	networking-istio-6cd7bcf9d-r6257	1/1	Running	0
knative-serving	webhook-77f754cc5c-mzhkg	1/1	Running	0
kube-flannel	kube-flannel-ds-cfflf	1/1	Running	0
kube-system	coredns-f9fd979d6-htkjs	1/1	Running	0
kube-system	coredns-f9fd979d6-zphlf	1/1	Running	0
kube-system	etcd-vk8s-1-19-ubuntu-node-0-0	1/1	Running	0
kube-system	ip-masq-agent-mz8s7	1/1	Running	0
kube-system	kube-apiserver-vk8s-1-19-ubuntu-node-0-0	1/1	Running	0
kube-system	kube-controller-manager-vk8s-1-19-ubuntu-node-0-0	1/1	Running	0
kube-system	kube-proxy-pqklh	1/1	Running	0
kube-system	kube-scheduler-vk8s-1-19-ubuntu-node-0-0	1/1	Running	0
kube-system	nvda-device-plugin-daemonset-tzjj6	1/1	Running	0
kubeflow-user-example-com	nl-pipeline-ui-artifact-75bd47846-xkglp	2/2	Running	0
kubeflow-user-example-com	nl-pipeline-visualizationserver-6fcd9886f-vcg6j	2/2	Running	0
kubeflow	admission-webhook-deployment-6998799f64-7s5pq	1/1	Running	0
kubeflow	cache-deployer-deployment-5bf84cf66f-9wshv	2/2	Running	2
kubeflow	cache-server-798b44675d-ggpbv	2/2	Running	0
kubeflow	centraldashboard-8cb996f694-74x88	1/1	Running	0
kubeflow	jupyter-web-app-deployment-d876474dc-qvp2f	1/1	Running	0
kubeflow	katib-controller-75c774846f-7ktgq	1/1	Running	0
kubeflow	katib-db-manager-798586cbc5-hkgwg	1/1	Running	2
kubeflow	katib-mysql-7454467486-sd4ch	1/1	Running	0
kubeflow	katib-ui-7c669ff554-2sfhp	1/1	Running	0

- vk8s + kubeflow 환경에서의 다양한 학습 수행 완료
  - Federation Learning(연합학습)
    - 연합학습은 기기나 기관 등이 여러 위치에 분산 저장된 데이터를 직접 공유하지 않고 서로 협력하며 인공지능 모델을 학습할 수 있는 분산형 학습 기법

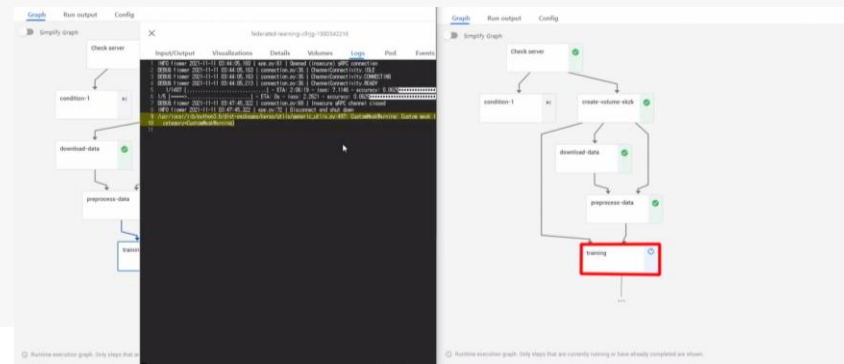


## GEDGE 환경에서 연합학습 실행 확인

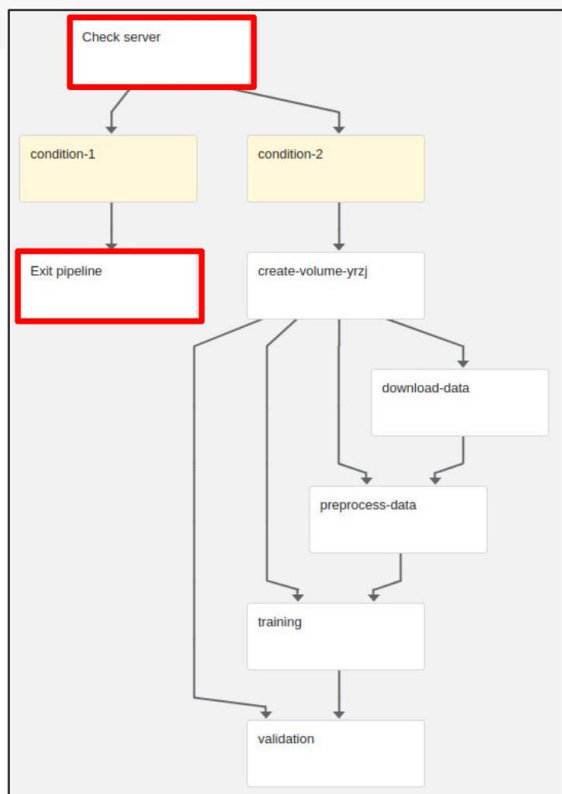
- ① On-demand로 GPU Clustering(vk8s) 환경을 자동으로 구성
- ② Infra as a Code로 kuberflow 용이하게 설치
- ③ Federated Learning으로 학습-추론 실행
- ④ 학습 시 반복되는 단계는 Kubeflow의 Pipeline으로 정의
- ⑤ Federated Learning 프레임워크는 Flower (<https://flower.dev/>)사용
- ⑥ 모델 : 이미지 분류모델인 EfficientNetB0 모델 사용
- ⑦ 데이터: cifar10 이미지 데이터 활용



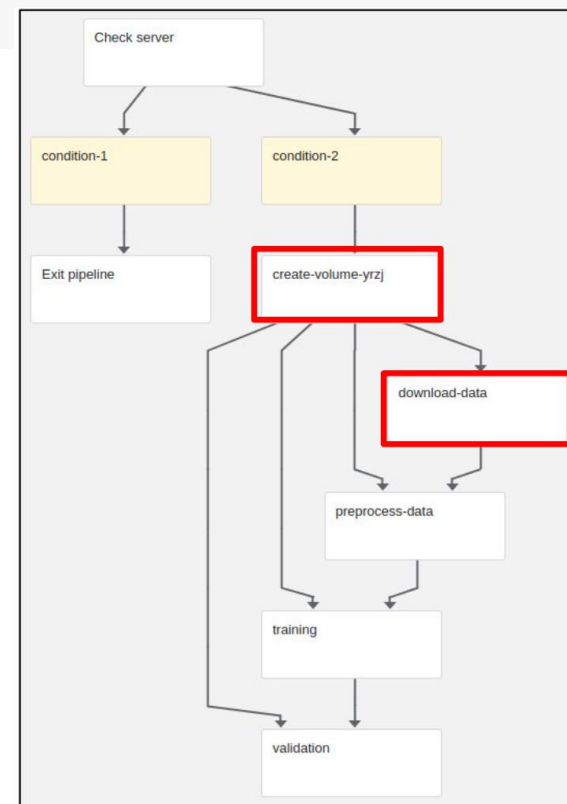
NAME	STATUS	ROLES	AGE	VERSION
master	Ready	control-plane,master	39h	v1.22.3
worker	Ready	<none>	39h	v1.22.3



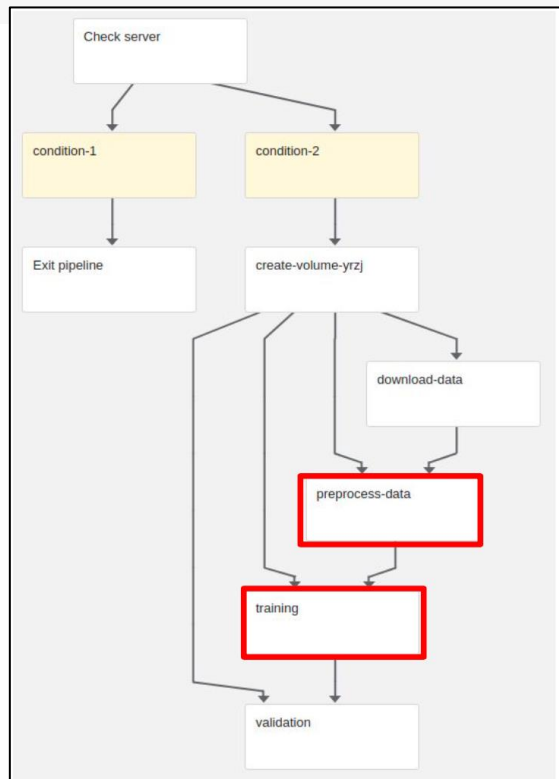
- Step 1. Check server
  - Federated Server가 실행 중인지 확인
- Step 2-1. Exit Pipeline
  - 파이프라인을 종료



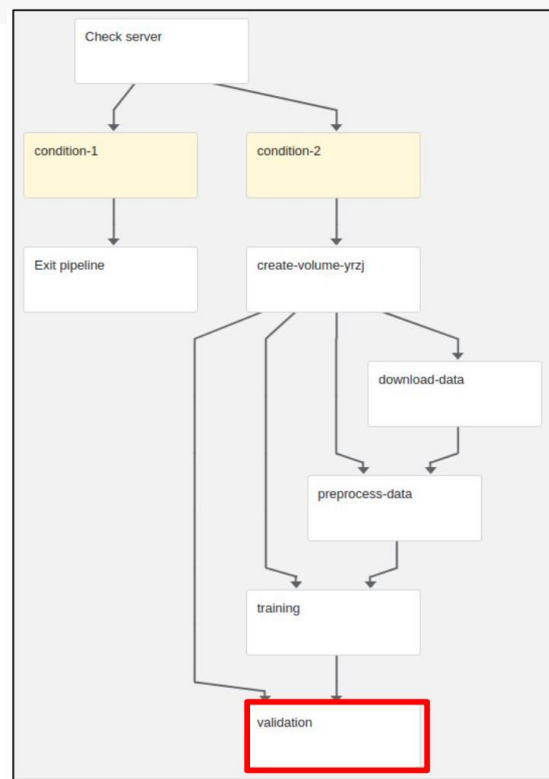
- Step 2-2. Create volume
  - 학습에 사용될 데이터와 모델 및 데이터 활용 지속성을 위해 볼륨 생성
- Step 3. Download data
  - 학습 이미지(cifar)를 다운로드 받아서 볼륨에 저장



- Step 4. Preprocess data
  - 다운로드 받은 이미지 데이터를 일정한 크기의 numpy 배열로 변환하고 다시 볼륨에 저장
- Step 5. Training
  - 전처리된 데이터로 학습을 진행

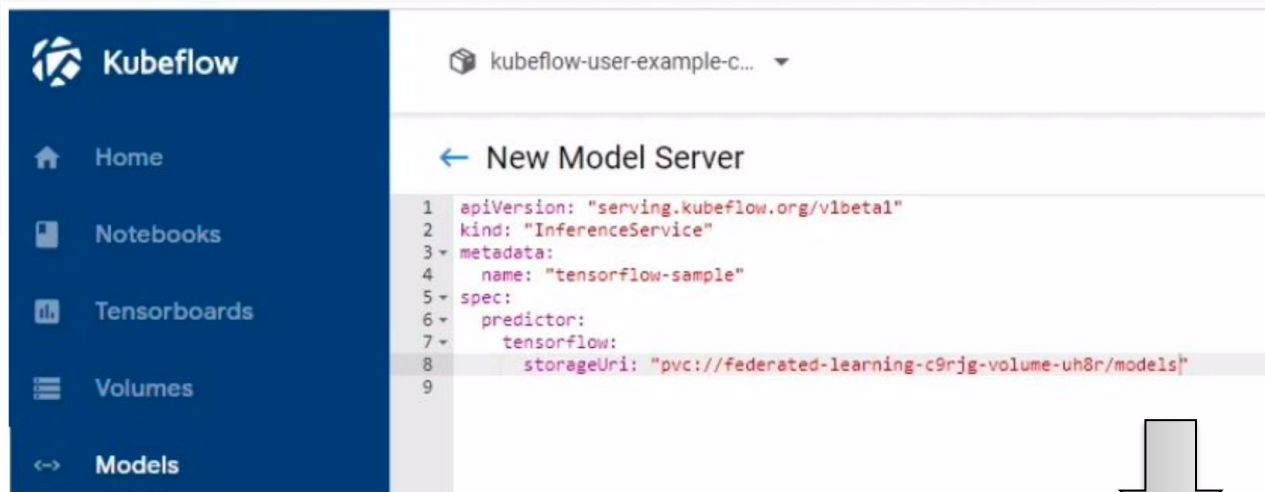


- Step 6.
  - 볼륨에 저장된 모델을 평가하여 모델이 제대로 저장됐는지 정확성을 잘 나오는지 검증

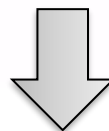






- Step 7.
  - Model Server 확인 및 결과 확인



```
1 apiVersion: "serving.kubeflow.org/v1beta1"
2 kind: "InferenceService"
3 metadata:
4   name: "tensorflow-sample"
5 spec:
6   predictor:
7     tensorflow:
8       storageUri: "pvc:///federated-learning-c9rjg-volume-uh8r/models/"
9
```



Model Servers							<a href="#">+ NEW MODEL SERVER</a>
Status	Name	Age	Predictor	Runtime	Protocol	Storage URI	
✓	tensorflow-sample	1 minute ago	Tensorflow	1.14.0		pvc:///federated-learning-c9rjg-volume-uh8r/mo...	 

```
(inner) root@sample-master:/serving$ curl localhost:8080/v1/models/tensorflow-sample:predict -d@./cat.json
{"predictions": [[3.70686593e-08, 4.95523234e-10, 4.13238467e-07, 0.999992967, 4.17128e-08, 6.1294736e-06, 2.40501151e-07, 2.33867254e-07, 3.50396978e-09, 1.03410986e-08]]}
```

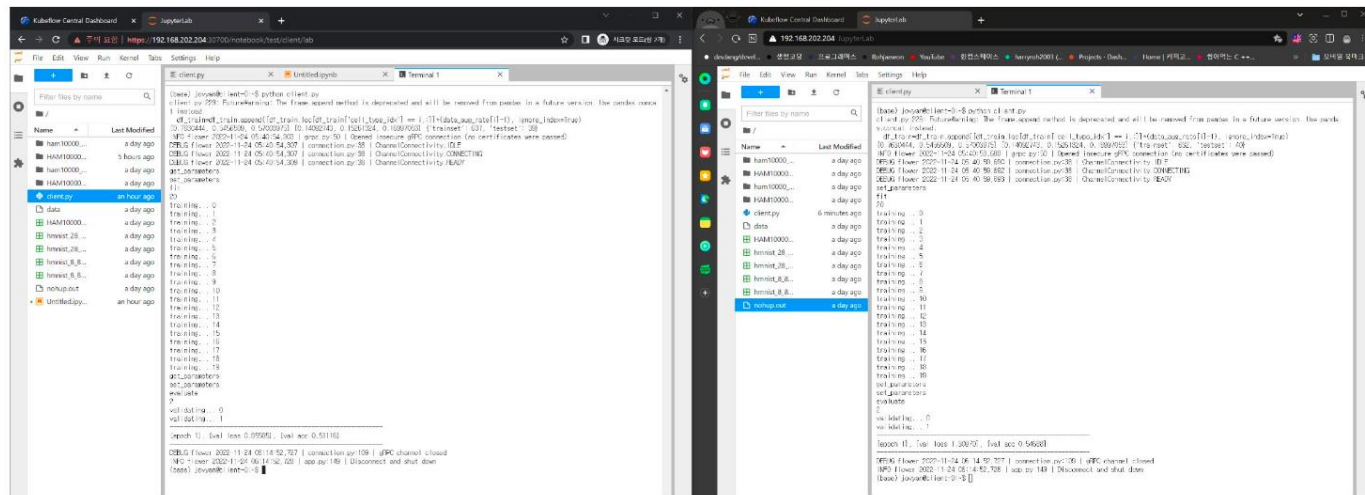


## ● 연합학습 실행 결과 확인

```
# DEVICE = torch.device("cuda:0" if torch.cuda.is_available() else "cpu")
load_data()
DEVICE = torch.device("cpu")
norm_mean, norm_std, df_train, df_val, num_examples = preprocess()
print(norm_mean, norm_std, num_examples)
model, input_size = load_model(DEVICE)
train_loader, val_loader = normalize(norm_mean, norm_std, df_train, df_val, input_size)
optimizer = optim.Adam(model.parameters(), lr=1e-3)
criterion = nn.CrossEntropyLoss().to(DEVICE)
epoch_num = 1
best_val_acc = 0
```

```
client = CifarClient(model, train_loader, val_loader, num_examples, epoch_num)
fl.client.start_numpy_client(server_address="192.168.202.204:32394", client=client)
```

클라이언트에 연합 학습 서버 엔드포인트 추가



- 사용자 별 노트북을 만들어 자신만의 데이터를 가지고 모델 학습 코드를 작성 및 실행 가능
- 모든 사용자들이 학습을 마치고 weight를 서버로 주면, 서버가 평균 weight를 가진 모델을 외부 저장소의 {group}/{project}/{round} 경로에 저장

- GEDGE 플랫폼에서는 본 컨퍼런스에서 발표한 Kubeflow의 방식이 아닌, 다양한 ML 학습-추론 실행 관리 툴 적용도 확장 적용 가능함
  - 컨테이너 기반의 Cloud 환경에서는 사용자 작업 환경을 이미지화하여 저장하고 재실행이 가능
- ML 학습-추론에 필요한 다양한 학습 데이터 및 모델 데이터는 GEDGE 플랫폼 저장소에 권한 설정, 저장 및 활용이 자유로움
  - 사용자별 저장 서비스 제공
  - 사용자가 생성한 컨테이너에 사용자 별 저장소를 자동으로 마운트하여 제공
- 본 컨퍼런스에서 발표한 연합학습 외에도 전이학습, 강화학습 등 다양한 학습이 가능함
  - GEDGE 플랫폼에서는 다양한 타입의 가속장치(GPU)를 탑재하고 있음
  - 컨테이너 기반으로 다양한 ML 플랫폼 및 ML 개발환경 제공

# 감사합니다.

<http://gedge-platform.github.io>



GS-Optops 커뮤니티 멤버  
조정현(junghyuncho@sk.com)

## Welcome to GEdge Platform

An Open Cloud Edge SW Platform to enable Intelligent Edge Service

### GEdge Platform will lead Cloud-Edge Collaboration