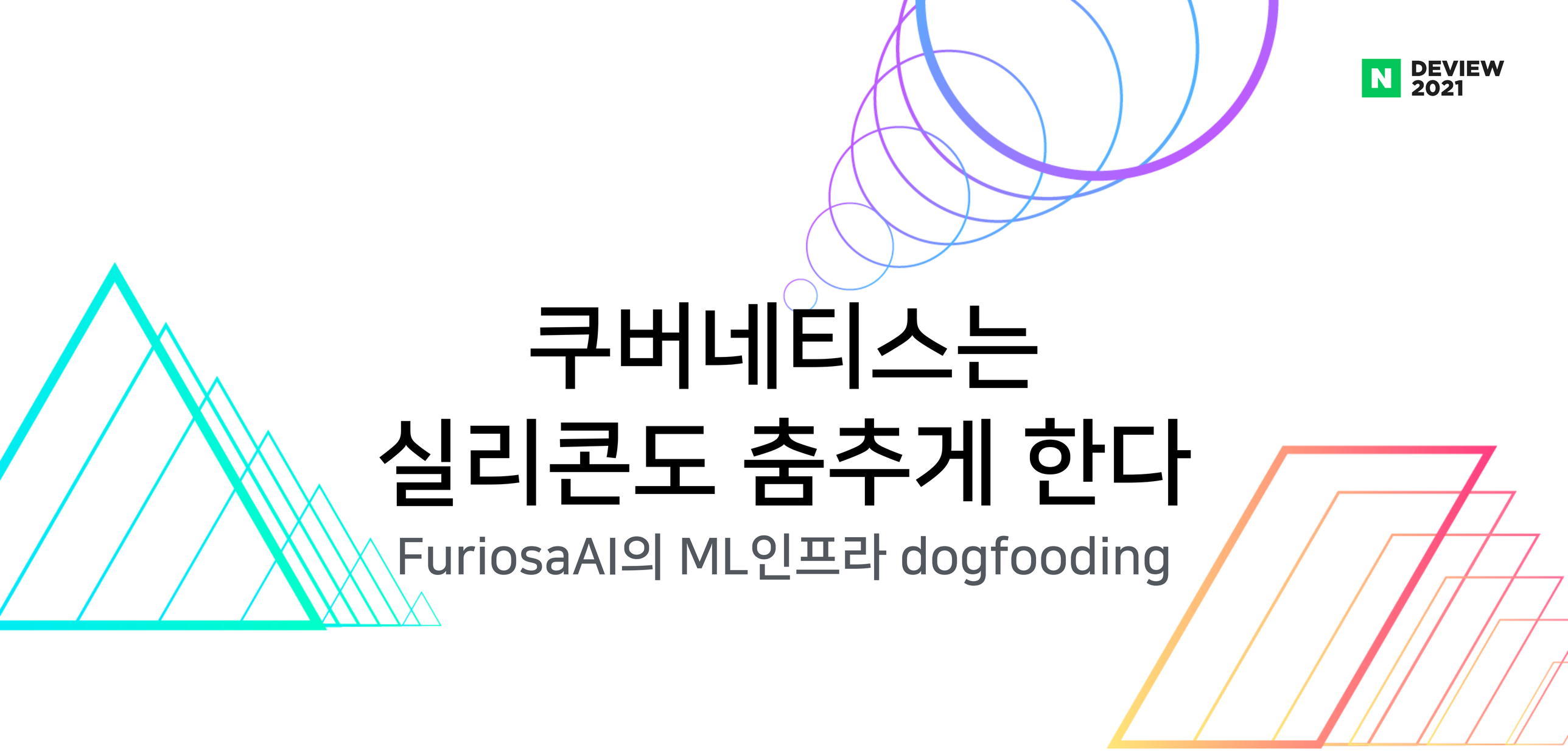


쿠버네티스는 실리콘도 춤추게 한다

FuriosaAI의 ML인프라 dogfooding

이병찬 FuriosaAI

DEVIEW 2021

About FuriosaAI

About speaker

이병찬
(github.com/EleganceLESS)

- FuriosaAI Software Engineer
- Platform Software Development
  - Software Development Kits
  - Toolkit (CLI, Integration, Observability, etc.)
- 더 아름다운 코드를 꿈꾸고 지향합니다.

CONTENTS

1.Motivation

1. Motivation

DEVIEW 2021

AI/ML Industry의 급격한 발전과 성장

- 데이터셋의 질적/양적 개선
- 알고리즘의 진화와 발전
- Cycle Iteration을 통한 지속적인 개선
- 더 강력해지는 컴퓨팅 자원 (스토리지, 네트워크 포함)

1. Motivation

DEVIEW 2021

GPU 그리고 NPU

- (in ML world,) GPU is a general purpose device
- 강력한 Software와 훌륭한 생태계
- 가격이 비싸고 수요가 많아 구하기 어려움

- NPU is a specialized device
- 목표가 명확한만큼, 보다 효율적으로 동작함

1. Motivation

어떻게 제공할 것인가?

- 사용자 개인에게 1대씩 개별 지급
- 서버에 다수를 장착 후 접속하여 사용
- 서버는 Hypervisor로 구성하고 사용자에게는 VM 할당
- Container 기반의 환경 제공

1. Motivation

DEVIEW 2021

Containerization 시 고려해야 할 점들

- Isolation
  어떻게 리소스를 격리시켜 독립적인 환경을 만들 것인가?
- Utilization
  어떻게 리소스를 분배하여 효율적으로 사용할 것인가?
- Abstraction
  어떻게 디테일을 숨기고 **편리한 사용자 경험**을 제공할 것인가?

2.k8s Device Plugin

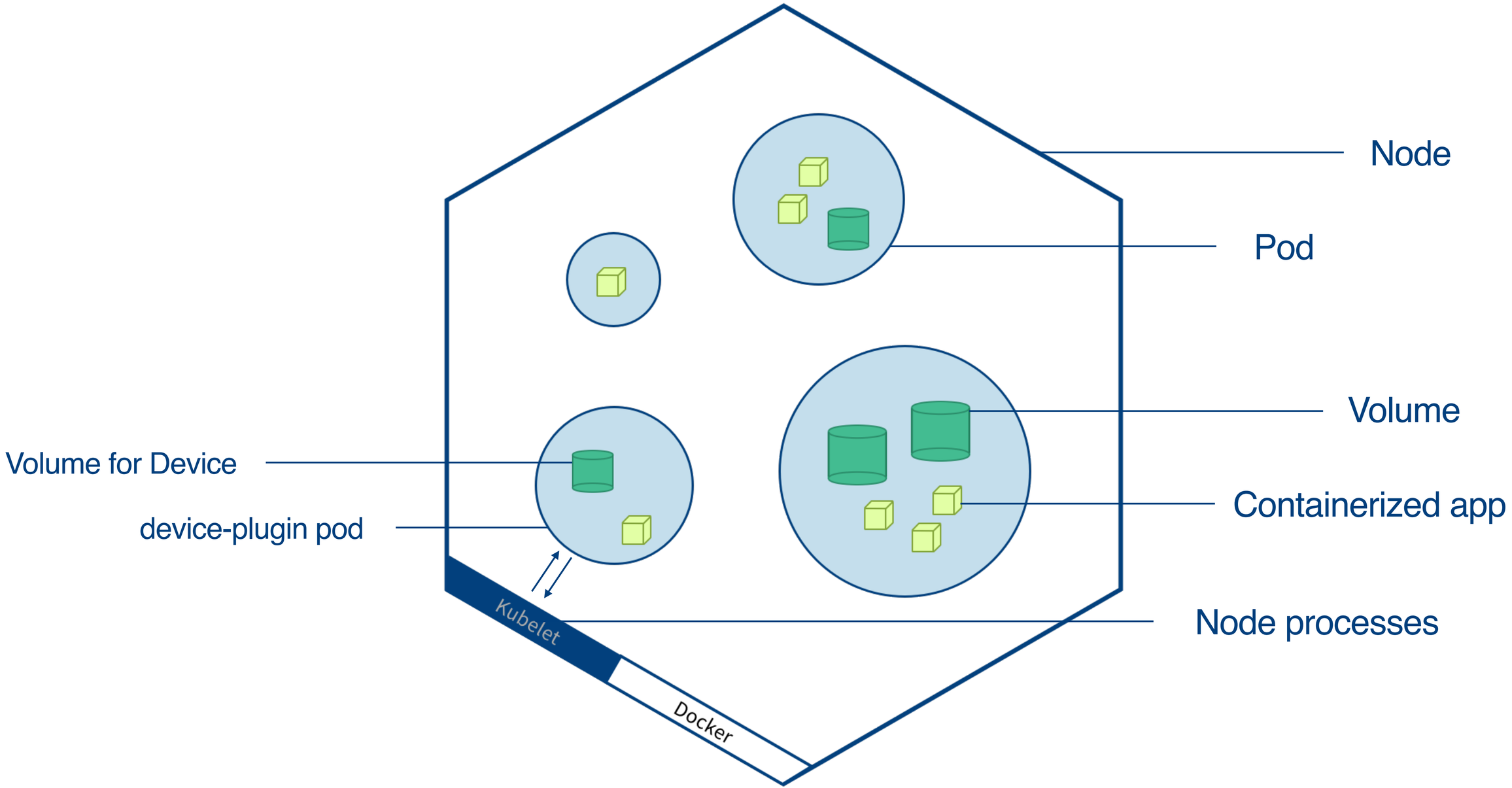
2. k8s-device-plugin

DEVIEW 2021

k8s에서 Hardware 가속기를 지원하기 위한 도구

- NVIDIA에서 제안
- “Extended Resource”에서 발전되었음
- k8s를 머신러닝, 딥러닝에도 충분히 활용하기 위한 설계
- 여러 Hardware 제품에 대한 플러그인이 제공되고 있음
    - AMD, Intel, NVIDIA GPU
    - RDMA
    - Xilinx FPGA
    - etc.

2.1 kubelet & device-plugin

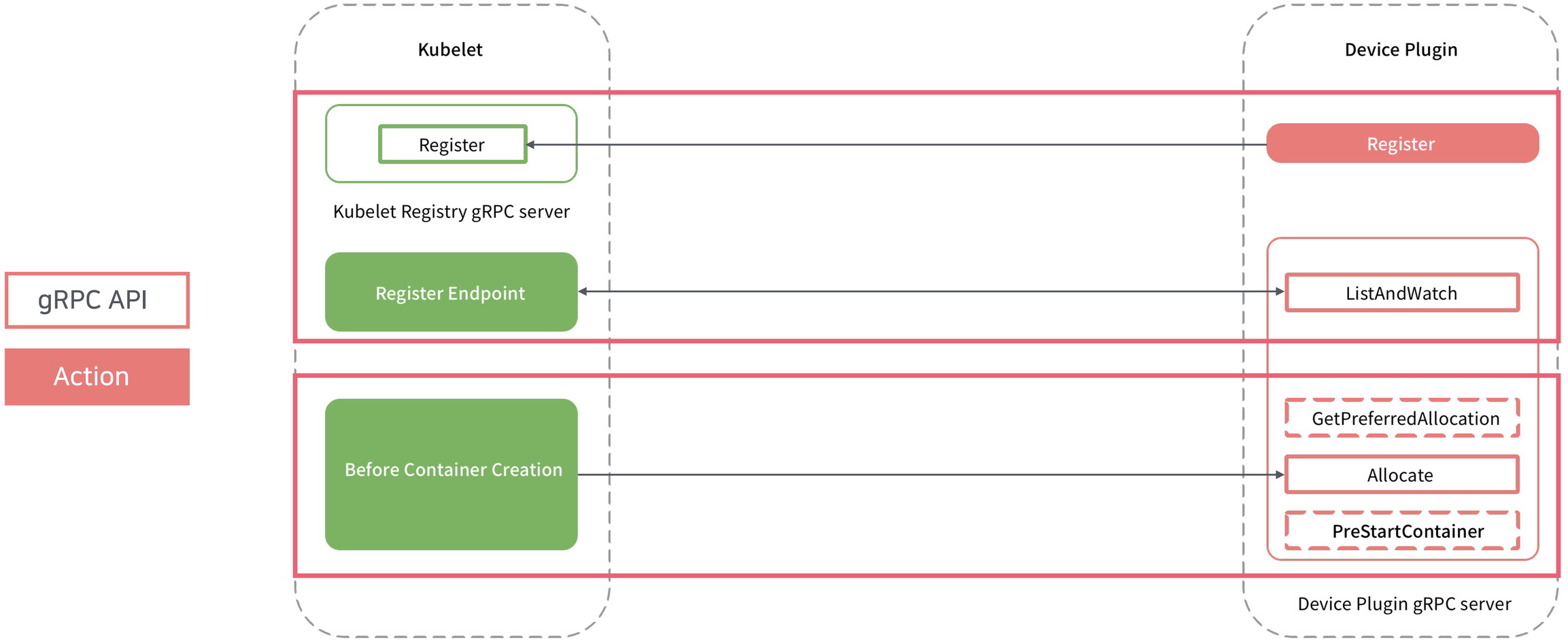


2.1 kubelet & device-plugin

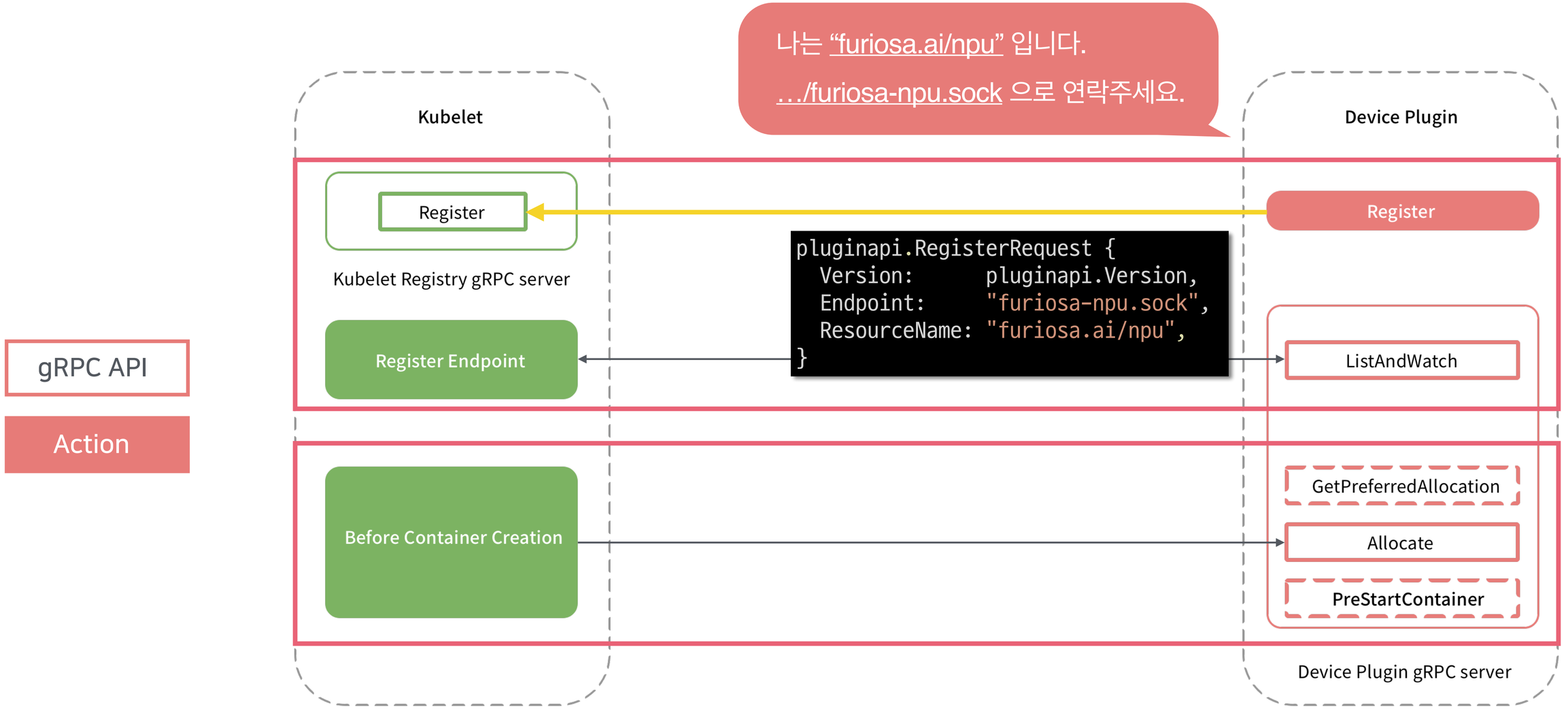
DEVIEW 2021

```
// kubelet/pkg/apis/deviceplugin/v1beta1/api.proto

service DevicePlugin {

  rpc GetDevicePluginOptions(Empty) returns (DevicePluginOptions) {}

  rpc ListAndWatch(Empty) returns (stream ListAndWatchResponse) {}

  rpc GetPreferredAllocation(PreferredAllocationRequest) returns (PreferredAllocationResponse) {}

  rpc Allocate(AllocateRequest) returns (AllocateResponse) {}

  rpc PreStartContainer(PreStartContainerRequest) returns (PreStartContainerResponse) {}

}
```

2.1 kubelet & device-plugin

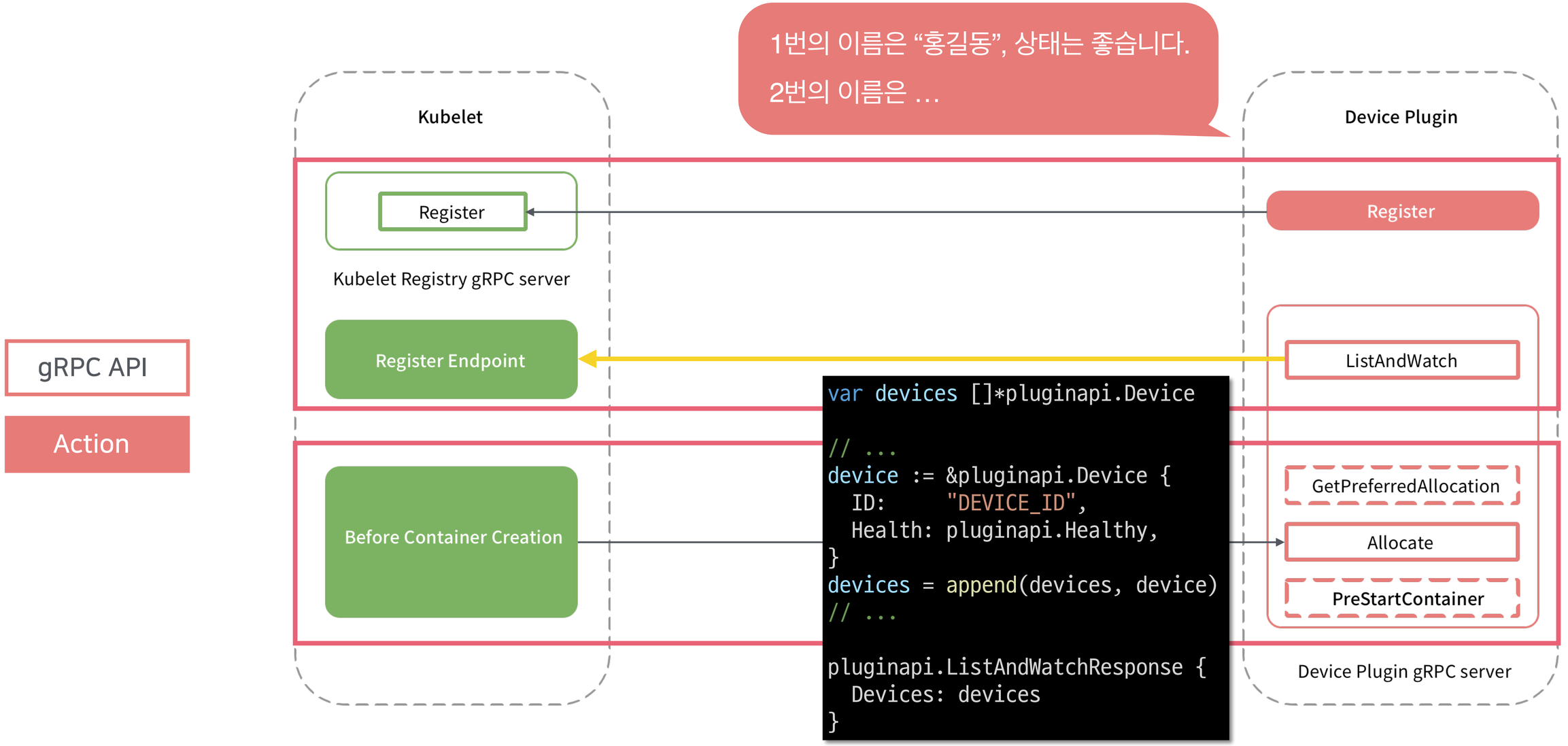


2.2 Registration

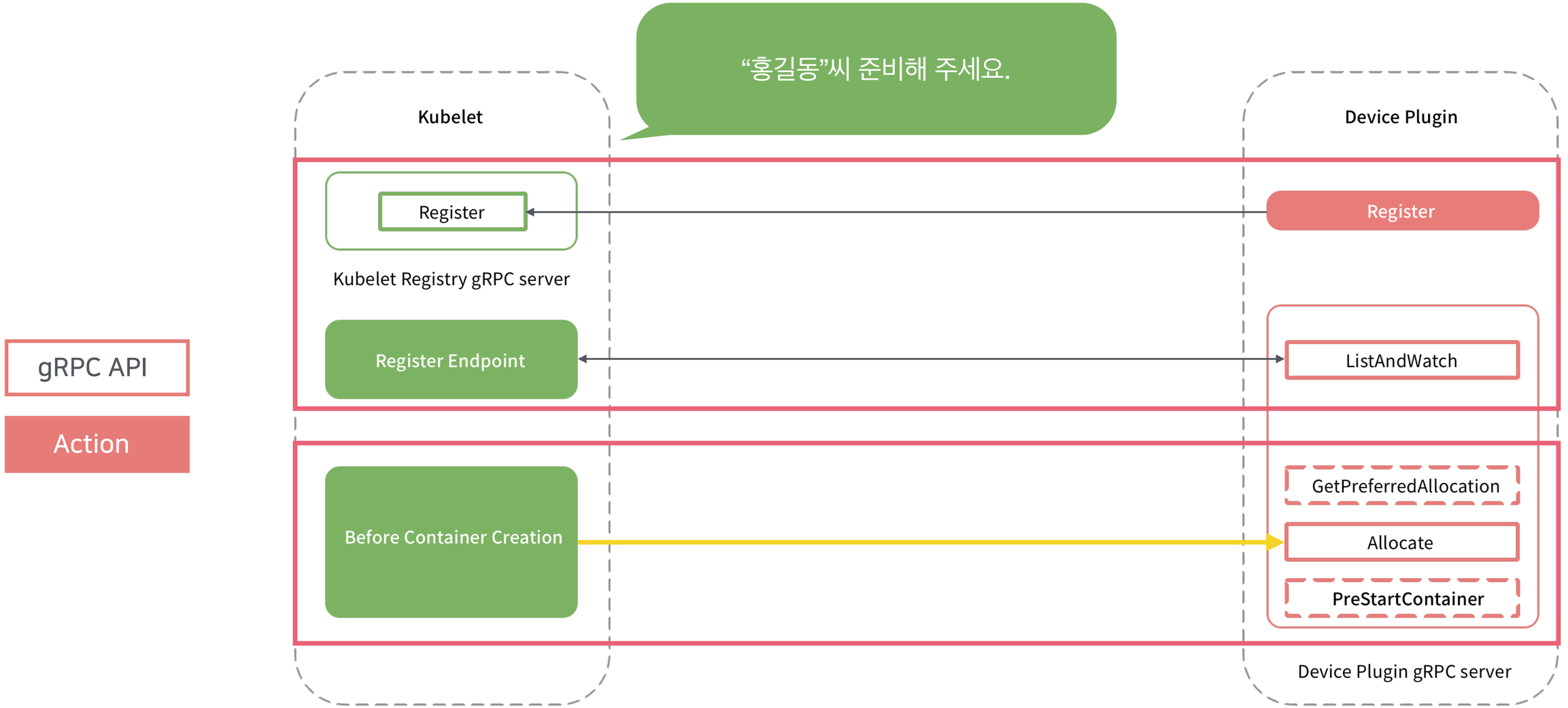


2.3 ListAndWatch

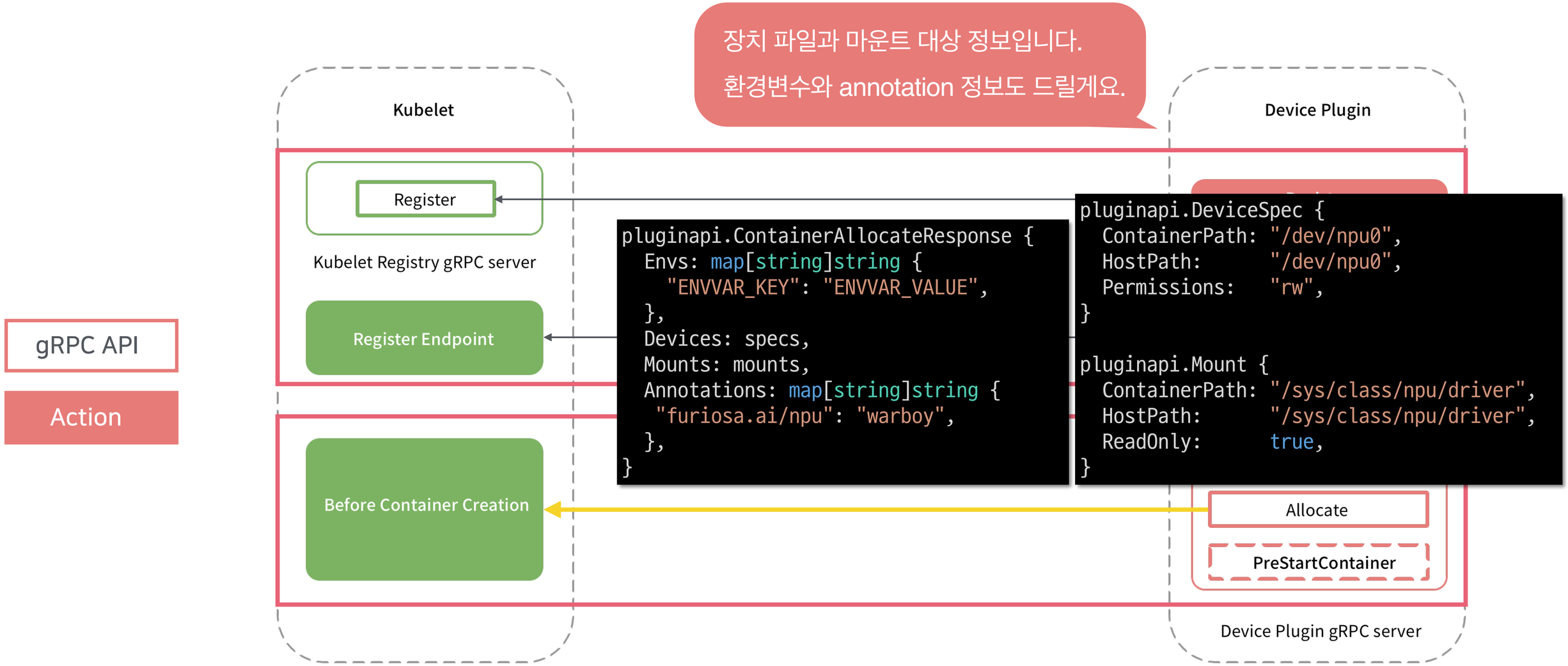
2.3 ListAndWatch



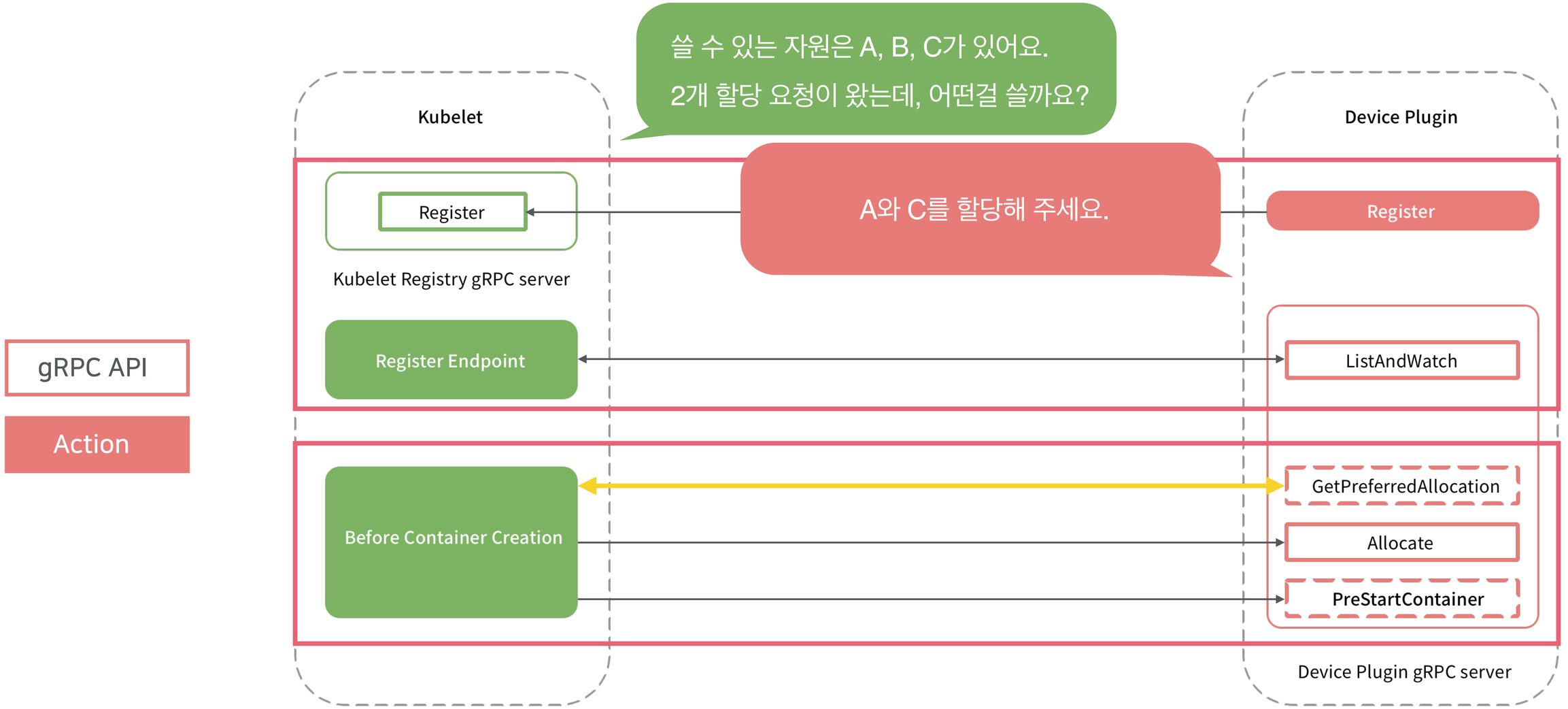
2.4 Allocation



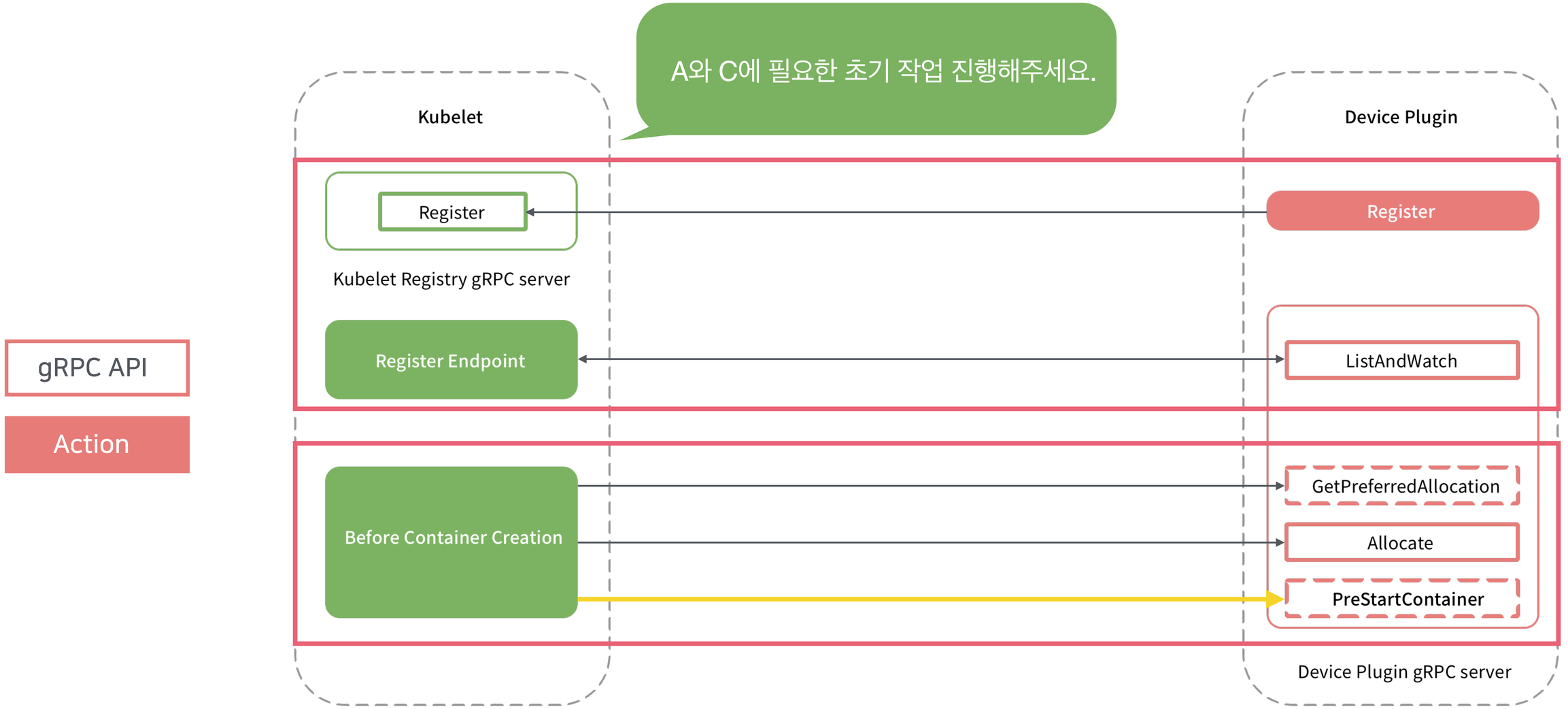
2.4 Allocation

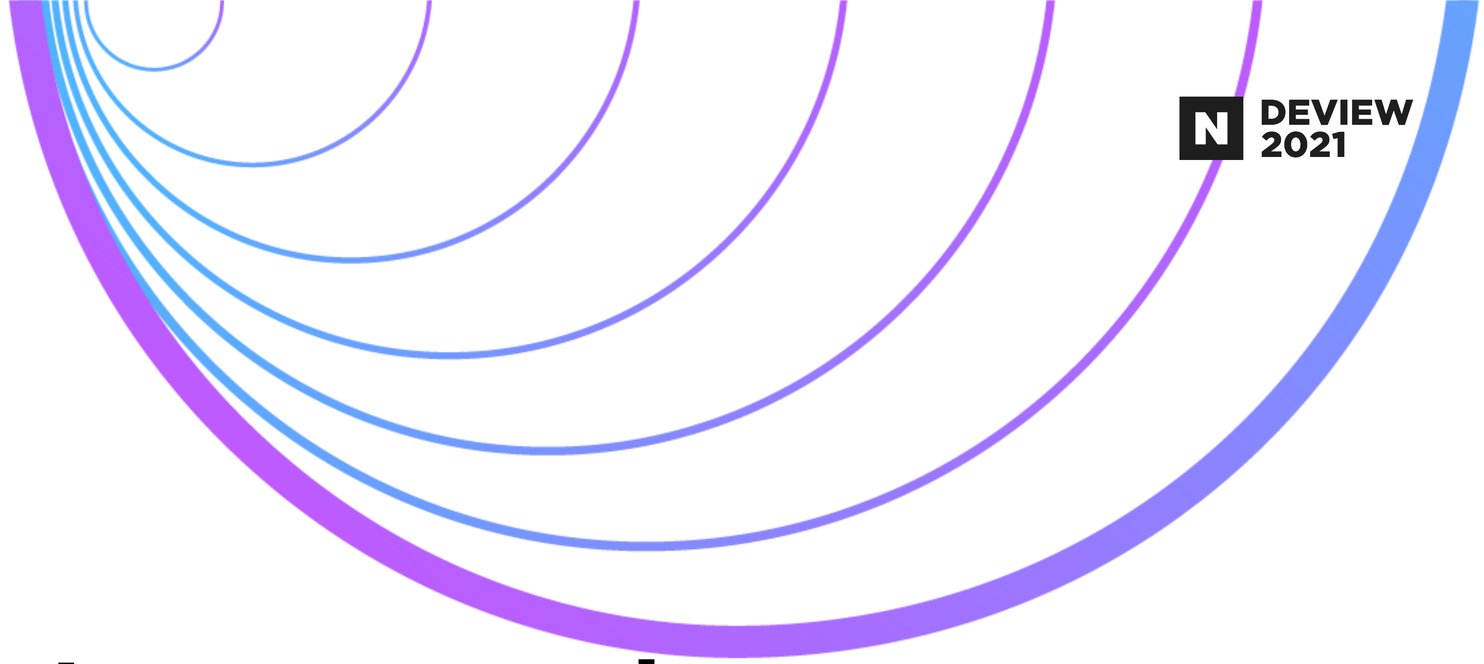


2.5 Optional APIs



2.5 Optional APIs





3.k8s Integration

3.1 설치 단계

DEVIEW 2021

k8s에서 GPU/NPU를 사용하기 위한 준비 과정

- Install Device Driver
- Install nvidia-docker

NVIDIA GPU

- Install Device Driver

FuriosaAI NPU

3.1 설치 단계

DEVIEW 2021

k8s에서 GPU/NPU를 사용하기 위한 준비 과정

github.com/NVIDIA/k8s-device-plugin

- Create DaemonSet
  : nvidia-device-plugin.yml

또는

- helm install

- Create DaemonSet
  : furiosa.ai/k8s-device-plugin
  : furiosa.ai/k8s-node-labeller

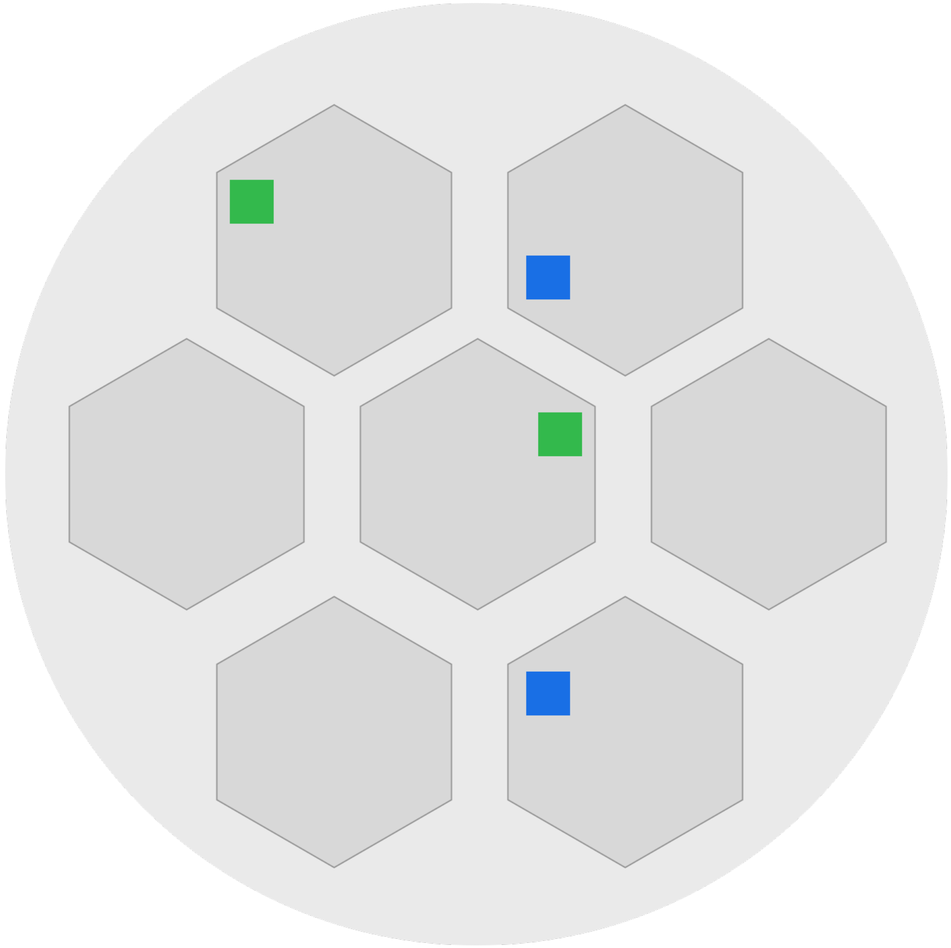
NVIDIA GPU

FuriosaAI NPU

3.1 설치 단계

DEVIEW 2021

k8s Cluster

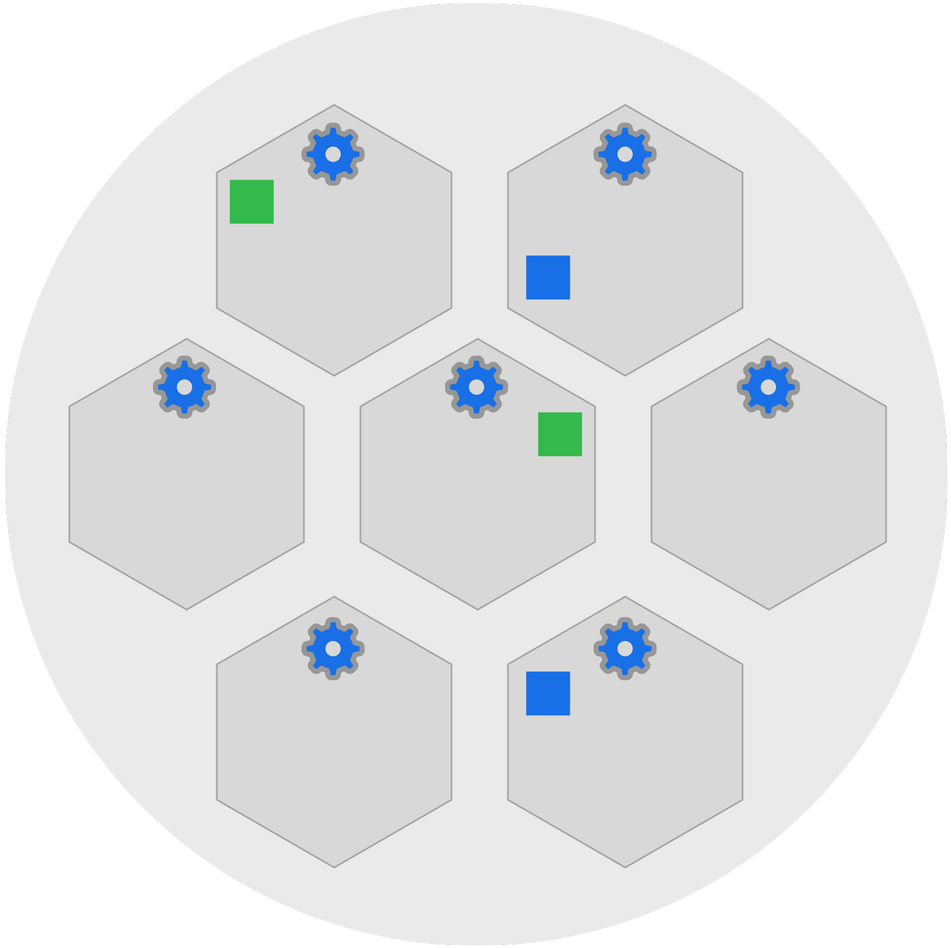


- NVIDIA GPU Device
- FuriosaAI NPU Device

3.1 설치 단계

DEVIEW 2021

Create DaemonSet



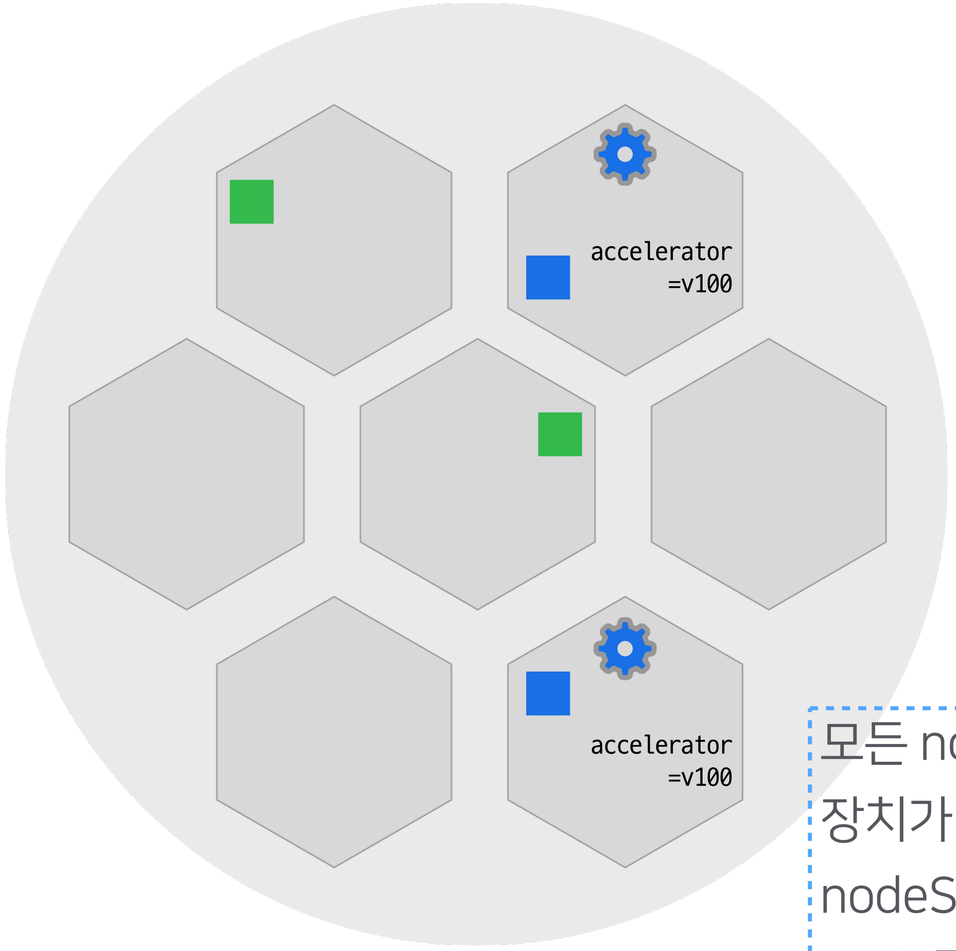
- NVIDIA GPU Device
- FuriosaAI NPU Device
- NVIDIA Device Plugin

DaemonSet을 생성하면
클러스터 내의 모든 node에
Device Plugin Pod이
생성되고 실행됩니다.

3.1 설치 단계

DEVIEW 2021

Create DaemonSet with nodeSelector



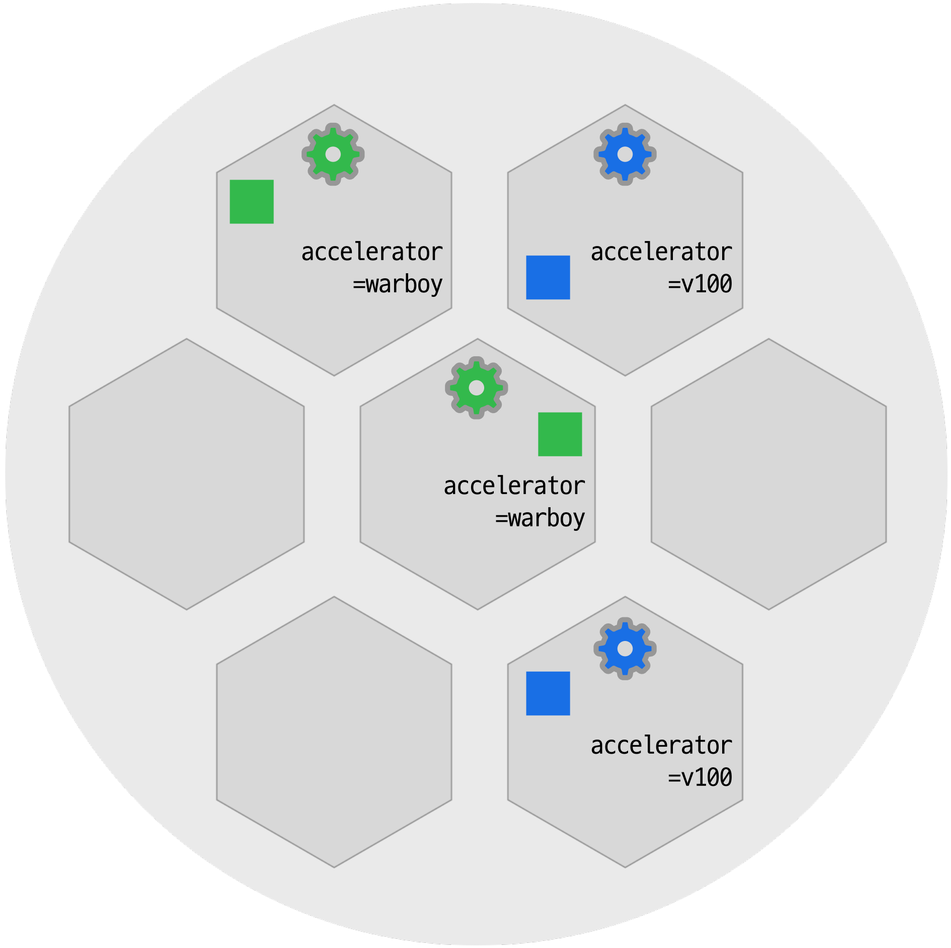
- NVIDIA GPU Device
- FuriosaAI NPU Device
- NVIDIA Device Plugin

모든 node에 실행될 필요는 없으므로, 장치가 있는 node에 label을 달고 nodeSelector로 DaemonSet을 설치할 node를 지정합니다.

3.1 설치 단계

DEVIEW 2021

Create DaemonSet with nodeSelector



- NVIDIA GPU Device
- FuriosaAI NPU Device
- NVIDIA Device Plugin
- FuriosaAI Device Plugin

3.1 설치 단계

DEVIEW 2021

Automatic Metadata Labelling

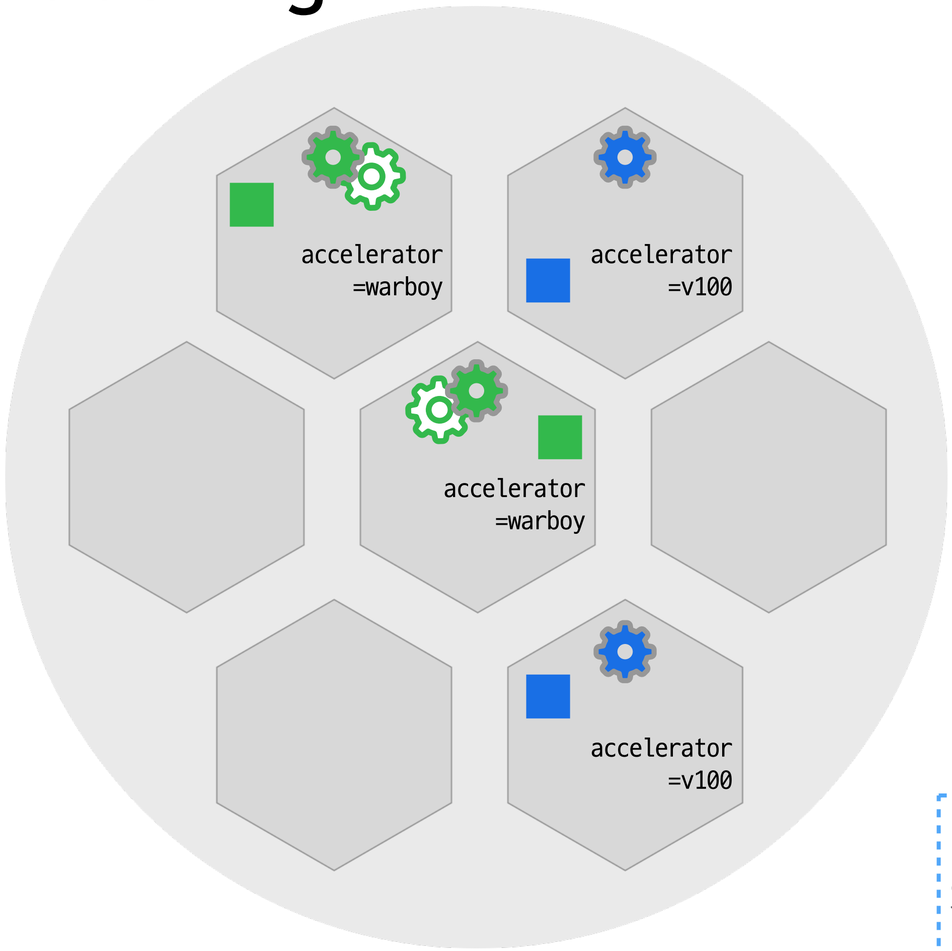
Device ID

Architecture Name

Family Name

Version

Spec…



- NVIDIA GPU Device
- FuriosaAI NPU Device
- NVIDIA Device Plugin
- FuriosaAI Device Plugin
- FuriosaAI Node Labeller

node-labeller를 통해
장치의 메타데이터 정보를 label에
자동 추가할 수 있습니다.

3.1 설치 단계

DEVIEW 2021

```
$ kubectl describe node gpu-server
```

```
Capacity:
  cpu:                40
  memory:             131663968Ki
  nvidia.com/gpu:     4
  pods:               110
Allocatable:
  cpu:                40
  memory:             131561568Ki
  nvidia.com/gpu:     4
  pods:               110


Allocated resources:
  (Total limits may be over 100 percent, i.e., overcommitted.)
  Resource            Requests      Limits
  --------            --------      ------
  cpu                 24550m (61%)  16 (40%)
  memory              97024Mi (75%)  84Gi (66%)
  nvidia.com/gpu      2             2
```

```
$ kubectl describe node npu-server
```

```
Capacity:
  cpu:                128
  furiosa.ai/npu:     5
  memory:             528277624Ki
  pods:               110
Allocatable:
  cpu:                128
  furiosa.ai/npu:     5
  memory:             528175224Ki
  pods:               110


Allocated resources:
  (Total limits may be over 100 percent, i.e., overcommitted.)
  Resource            Requests   Limits
  --------            --------   ------
  cpu                 250m (0%)  0 (0%)
  memory              0 (0%)     0 (0%)
  furiosa.ai/npu      4          4
```

NVIDIA GPU

FuriosaAI NPU

3.2 Pod

gpu-pod.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: gpu-pod
spec:
  containers:
  - image: ubuntu:focal
    name: gpu-pod
    resources:
      limits:
        nvidia.com/gpu: "1"
      requests:
        nvidia.com/gpu: "1"
```

npu-pod.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: npu-pod
spec:
  containers:
  - image: ubuntu:focal
    name: npu-pod
    resources:
      limits:
        furiosa.ai/npu: "1"
      requests:
        furiosa.ai/npu: "1"
```

GPU/NPU를 리소스로 지정하여
Pod을 생성하는 예제입니다.

NVIDIA GPU

FuriosaAI NPU

3.2 Pod

DEVIEW 2021

```
$ kubectl apply -f gpu-pod.yaml
$ kubectl exec -it gpu-pod -- bash

# nvidia-smi

...

# /workspace/run_training.py
```

```
$ kubectl apply -f npu-pod.yaml
$ kubectl exec -it npu-pod -- bash

# furiosactl list

...

# /workspace/run_inference_batch.py
```

Pod내에서 GPU/NPU를 사용할 수 있음을 확인할 수 있습니다.

NVIDIA GPU

FuriosaAI NPU

3.3 Job

gpu-job.yaml

```yaml
apiVersion: v1
kind: Job
metadata:
  name: gpu-job
spec:
  template:
    spec:
      containers:
      - image: ubuntu:focal
        name: gpu-job
        resources:
          limits:
            nvidia.com/gpu: "1"
          requests:
            nvidia.com/gpu: "1"
        command: ["/workspace/scripts/run_training.sh"]
```

npu-job.yaml

```yaml
apiVersion: v1
kind: Job
metadata:
  name: npu-job
spec:
  template:
    spec:
      containers:
      - image: ubuntu:focal
        name: npu-job
        resources:
          limits:
            furiosa.ai/npu: "1"
          requests:
            furiosa.ai/npu: "1"
        command: ["/workspace/scripts/run_inference_batch.sh"]
```

GPU/NPU를 리소스로 지정하여
Job을 생성하는 예제입니다.

NVIDIA GPU

FuriosaAI NPU

3.4 CronJob

gpu-cron-job.yaml

```yaml
apiVersion: v1
kind: CronJob
metadata:
  name: gpu-cron-job
spec:
  schedule: "0 0 * * *"
  jobTemplate:
    spec:
      template:
        spec:
          containers:
          - image: ubuntu:focal
            name: gpu-cron-job
            resources:
              limits:
                nvidia.com/gpu: "1"
              requests:
                nvidia.com/gpu: "1"
            command: ["/workspace/scripts/run_training.sh"]
```

npu-cron-job.yaml

```yaml
apiVersion: v1
kind: CronJob
metadata:
  name: npu-cron-job
spec:
  schedule: "0 12 * * *"
  jobTemplate:
    spec:
      template:
        spec:
          containers:
          - image: ubuntu:focal
            name: npu-cron-job
            resources:
              limits:
                furiosa.ai/npu: "1"
              requests:
                furiosa.ai/npu: "1"
            command: ["/workspace/scripts/run_inference_batch.sh"]
```

GPU/NPU를 리소스로 지정하여
CronJob을 생성하는 예제입니다.

NVIDIA GPU

FuriosaAI NPU

3.5 Taint & Toleration

DEVIEW 2021

```
$ kubectl taint nodes gpu-server \
    nvidia.com/gpu=exclusive:NoSchedule

$ kubectl describe node gpu-server
```

```
Name:               gpu-server
Roles:              <none>
Labels:             beta.kubernetes.io/arch=amd64
                    beta.kubernetes.io/os=linux
                    kubernetes.io/arch=amd64
                    kubernetes.io/hostname=gpu-server
                    kubernetes.io/os=linux
CreationTimestamp:  Wed, 28 Jul 2021 19:55:36 +0900
Taints:             nvidia.com/gpu=exclusive:NoSchedule
```

```
$ kubectl taint nodes npu-server \
    furiosa.ai/npu=exclusive:NoSchedule

$ kubectl describe node npu-server
```

```
Name:               npu-server
Roles:              <none>
Labels:             beta.kubernetes.io/arch=amd64
                    beta.kubernetes.io/os=linux
                    kubernetes.io/arch=amd64
                    kubernetes.io/hostname=npu-server
                    kubernetes.io/os=linux
CreationTimestamp:  Wed, 28 Jul 2021 19:55:36 +0900
Taints:             furiosa.ai/npu=exclusive:NoSchedule
```

GPU/NPU가 장착된 node에 의도하지 않은 pod이 생성되는 것을 방지하기 위해 taint를 겁니다.

NVIDIA GPU

FuriosaAI NPU

3.5 Taint & Toleration

DEVIEW 2021

gpu-pod.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: gpu-pod
spec:
  tolerations:
  - key: "nvidia.com/gpu"
    value: "exclusive"
    operator: "Equal"
    effect: "NoSchedule"
  containers:
  - image: ubuntu:focal
    name: gpu-pod
    resources:
      limits:
        nvidia.com/gpu: "1"
      requests:
        nvidia.com/gpu: "1"
```

npu-pod.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: npu-pod
spec:
  tolerations:
  - key: "furiosa.ai/npu"
    value: "exclusive"
    operator: "Equal"
    effect: "NoSchedule"
  containers:
  - image: ubuntu:focal
    name: npu-pod
    resources:
      limits:
        furiosa.ai/npu: "1"
      requests:
        furiosa.ai/npu: "1"
```

taint를 걸어둔 node에서 pod을 생성하기 위해 toleration 옵션을 넣어줍니다.

NVIDIA GPU

FuriosaAI NPU

DEVIEW 2021

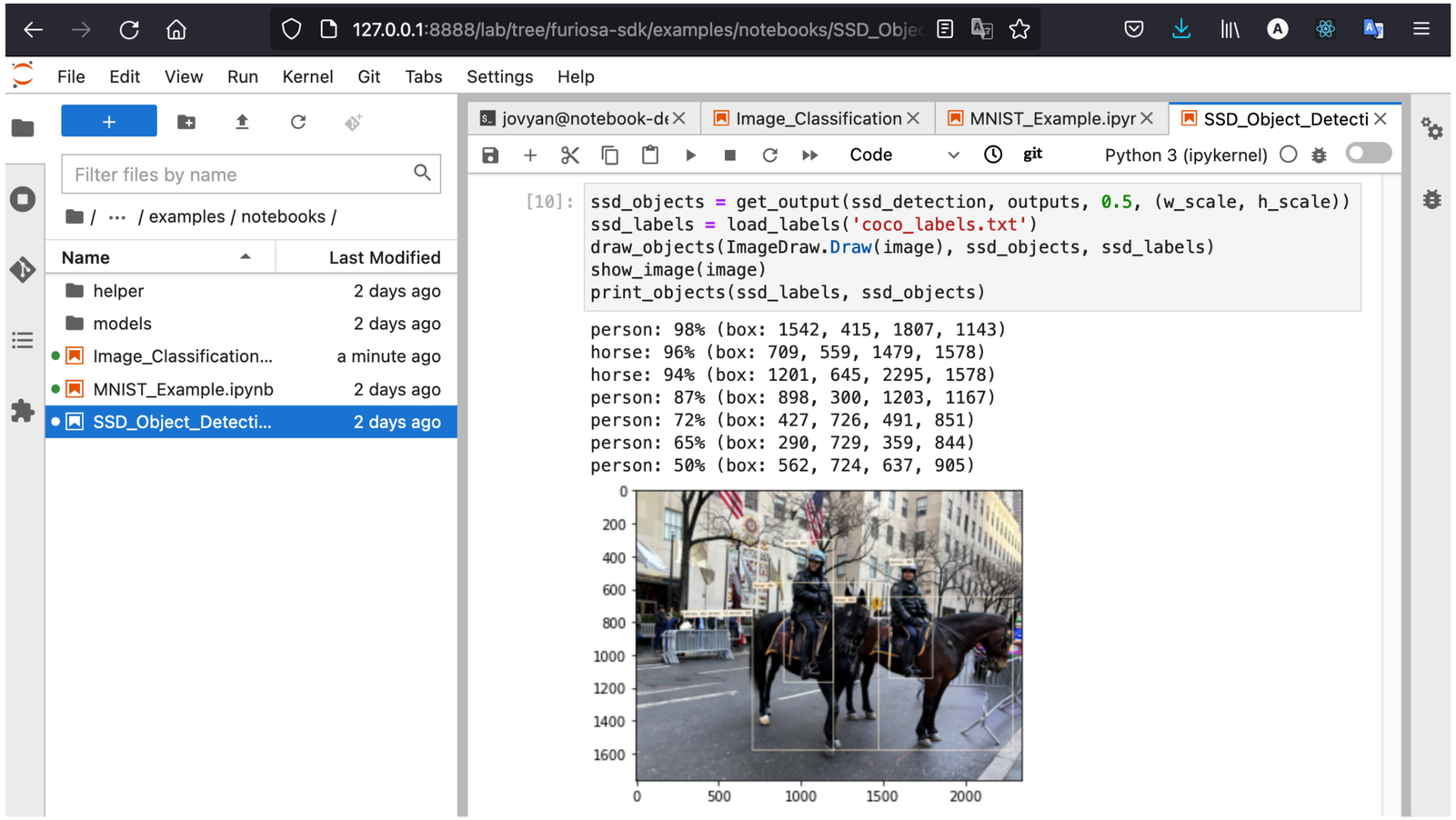
4.Use-case

4.1 JupyterLab

DEVIEW 2021

jupyter.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: jupyter-lab
spec:
  containers:
  - image: jupyter/base-notebook:ubuntu-20.04
    name: jupyter-lab
    env:
      - name: JUPYTER_ENABLE_LAB
        value: "yes"
    resources:
      limits:
        cpu: "4"
        memory: "8Gi"
        furiosa.ai/npu: "1"
      requests:
        cpu: "4"
        memory: "8Gi"
        furiosa.ai/npu: "1"
```



NPU하나를 할당하여 JupyterLab을 실행하는 예제입니다.

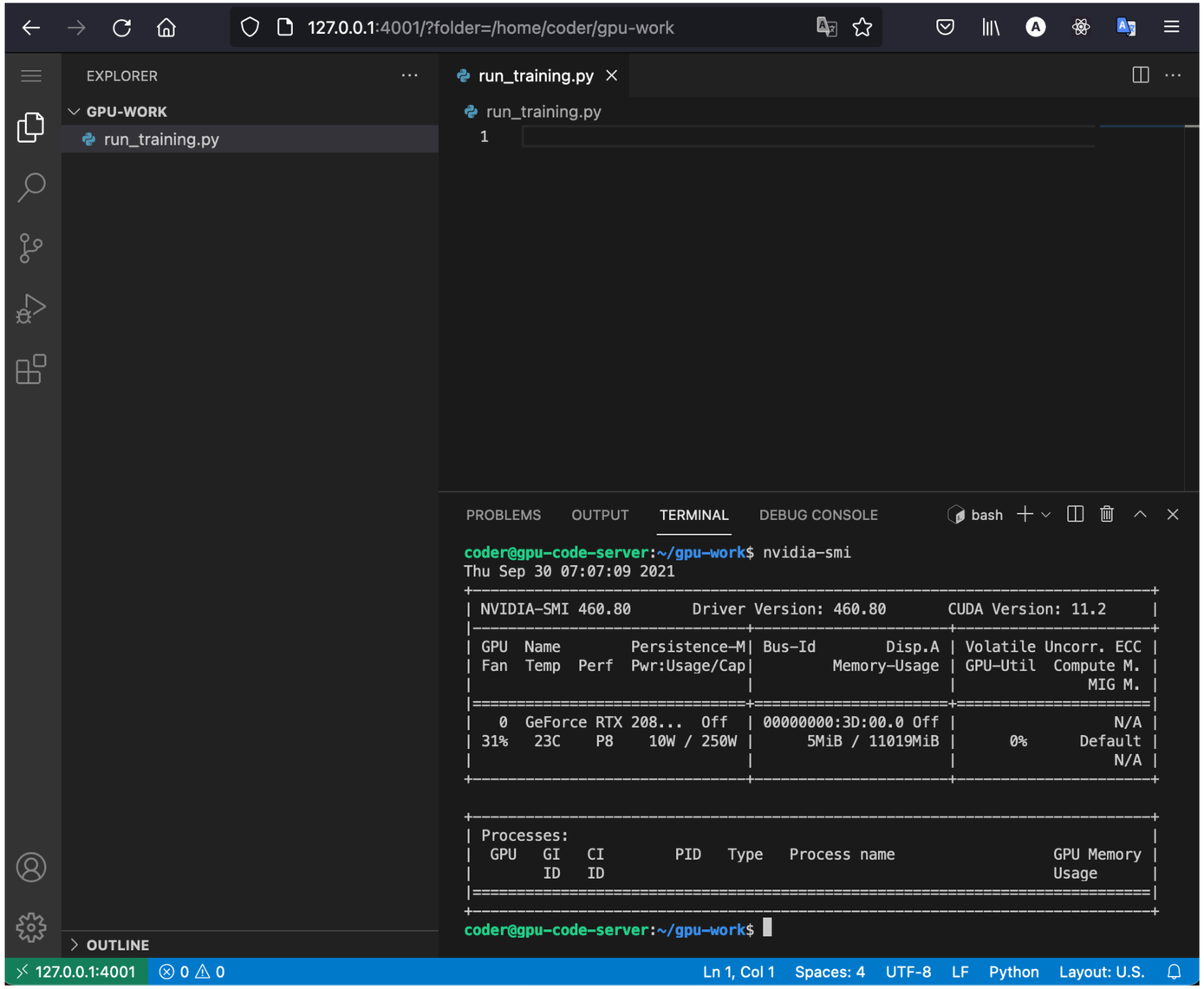
4.2 code-server

DEVIEW 2021

code-server.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: gpu-code-server
spec:
  containers:
  - image: codercom/code-server
    name: gpu-code-server
    resources:
      limits:
        cpu: "4"
        memory: "8Gi"
        nvidia.com/gpu: "1"
      requests:
        cpu: "4"
        memory: "8Gi"
        nvidia.com/gpu: "1"
```

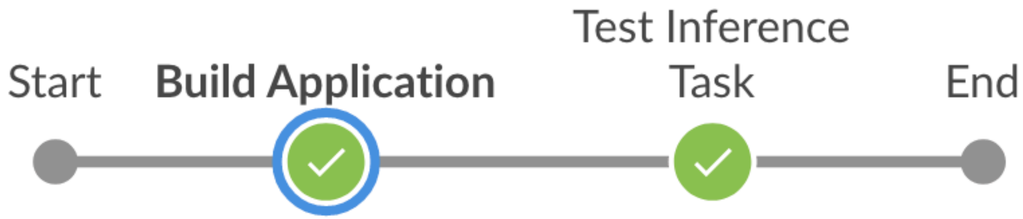
GPU하나를 할당하여 VS Code를 실행하는 예제입니다.



4.3 CI/CD (Jenkins)

Jenkinsfile

```
pipeline {
  stages {
    stage('Build Application') {
      agent { kubernetes {
        cloud "k8s-cluster"
        yaml cpuPod
      } }
      steps {
        container('cpu-pod') {
          // build application
        }
      }
    }
    stage('inference test') {
      agent { kubernetes {
        cloud "k8s-cluster"
        yaml npuPod
      } }
      steps {
        container('npu-pod') {
          // do Something with NPU
        }
      }
    }
  }
}
```

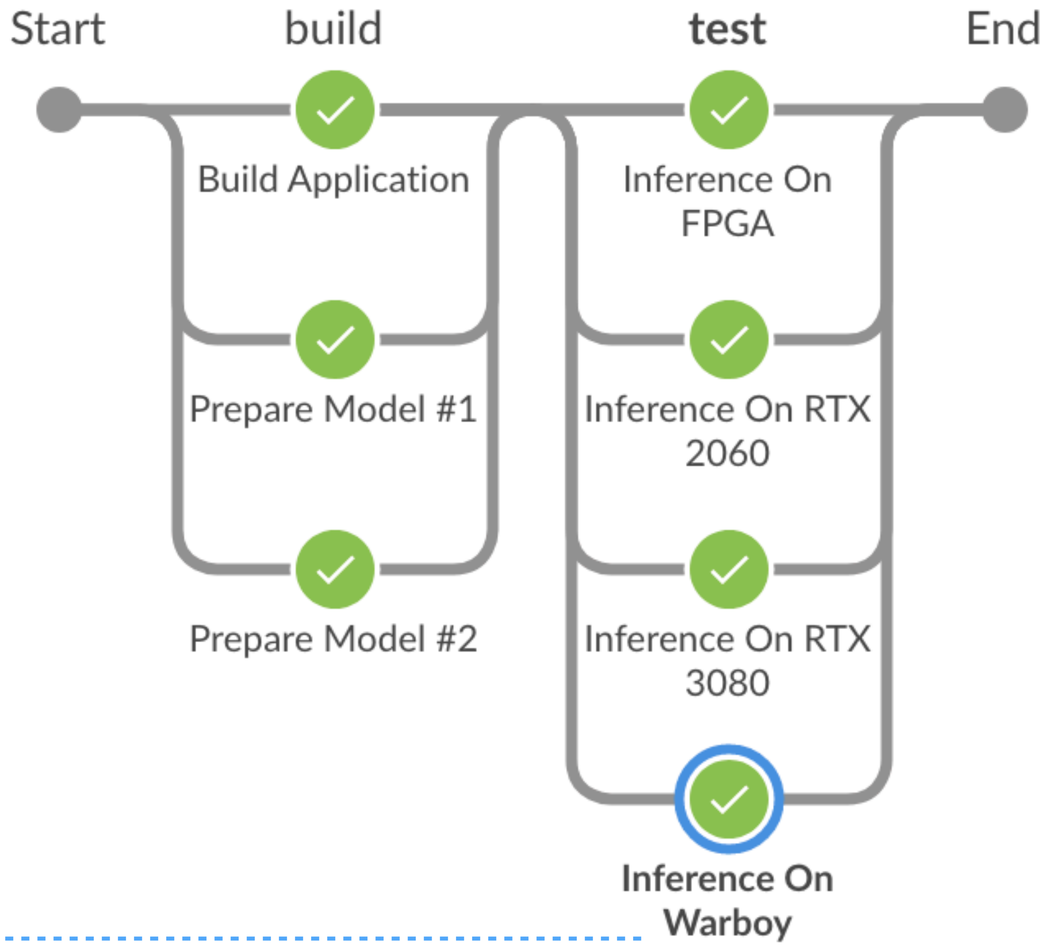


GPU/NPU pod을 이용하는 Task를
Jenkins Pipeline으로 구성하는 예제입니다.

4.3 CI/CD (Jenkins)

Jenkinsfile

```
pipeline {
  // ...
  stages {
    stage('build') {
      parallel {
        stage('Build Application') {
          // ...
        }
      }
    }
    stage('test') {
      parallel {
        stage('Inference On Warboy') {
          // ...
        }
        stage('Inference On FPGA') {
          // ...
        }
      }
    }
  }
}
```



GPU/NPU pod을 이용하는 Task를
Jenkins Pipeline으로 구성하는 예제입니다.

4.4 Application Deployment

DEVIEW 2021

application.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: awesome-application
  labels:
    app: awesome-application
spec:
  template:
    spec:
      containers:
      - image: some-company/awesome
        name: awesome-application
        ports:
        - containerPort: 8080
        resources:
          limits:
            cpu: "4"
            memory: "8Gi"
            furiosa.ai/npu: "1"
          requests:
            cpu: "4"
            memory: "8Gi"
            furiosa.ai/npu: "1"
```

```
furiosa-server --model-name mnist --model-path samples/data/MNIST_inception_v3_quant.tflite --
model-version 1
libnpu.so --- v2.0, built @ 52d60a8
INFO:furiosa.runtime.compiler:Saving the compiler log into /root/.local/state/furiosa/logs/
compile-20211013061141-v05gac.log
[1/6] 🔍   Compiling from tflite to dfg
[2/6] 🔍   Compiling from dfg to ldfg
▪▪▪▪▪ [1/3] Splitting graph...Done
▪▪▪▪▪ [2/3] Lowering...Done
▪▪▪▪▪ [3/3] Precalculating operators...Done
[3/6] 🔍   Compiling from ldfg to cdfg
[4/6] 🔍   Compiling from cdfg to gir
[5/6] 🔍   Compiling from gir to lir
[6/6] 🔍   Compiling from lir to enf
✨  Finished
Oct 13 06:14:46.076  INFO Npu (npu2pe0) is being initialized
Oct 13 06:14:46.079  INFO NuxInner create with pes: [PeId(0)]
Oct 13 06:14:46.239  INFO [Profiler] Program binary notification has been arrived. Cleanup current
profile queue data
INFO:     Started server process [3780]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://0.0.0.0:8080 (Press CTRL+C to quit)
```

NPU를 사용하는 AI Application을 서비스로 배포하는 예제입니다.

4.4 Application Deployment

DEVIEW 2021

```
POST /v2/models/mnist/versions/1/infer

{
  "inputs": [{
    "name": "mnist",
    "datatype": "INT32",
    "shape": [1, 1, 28, 28],
    "data": [ ...
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   3,  18,  18,  18, 126, 136, 175,  26, 166, 255, 247, 127,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,  30,  36,  94, 154, 170, 253, 253, 253, 253, 253, 225, 172, 253, 242, 195,  64,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,  49, 238, 253, 253, 253, 253, 253, 253, 253, 253, 251,  93,  82,  82,  56,  39,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,  18, 219, 253, 253, 253, 253, 253, 198, 182, 247, 241,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,  80, 156, 107, 253, 253, 205,  11,   0,  43, 154,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,   0,  14,   1, 154, 253,  90,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0, 139, 253, 190,   2,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  11, 190, 253,  70,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  35, 241, 225, 160, 108,   1,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  81, 240, 253, 253, 119,  25,   0,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  45, 186, 253, 253, 150,  27,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  16,  93, 252, 253, 187,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0, 249, 253, 249,  64,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  46, 130, 183, 253, 253, 207,   2,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  39, 148, 229, 253, 253, 253, 250, 182,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  24, 114, 221, 253, 253, 253, 253, 201,  78,   0,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,  23,  66, 213, 253, 253, 253, 253, 198,  81,   2,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,  18, 171, 219, 253, 253, 253, 253, 195,  80,   9,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,  55, 172, 226, 253, 253, 253, 253, 244, 133,  11,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0, 136, 253, 253, 253, 212, 135, 132,  16,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
        ... ]
  }]
}
```

4.4 Application Deployment

DEVIEW 2021

```
POST /v2/models/mnist/versions/1/infer

{
  "inputs": [{
    "name": "mnist",
    "datatype": "INT32",
    "shape": [1, 1, 28, 28],
    "data": [ ...
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   3,  18,  18,  18, 126, 136, 175,  26, 166, 255, 247, 127,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,  30,  36,  94, 154, 170, 253, 253, 253, 253, 253, 225, 172, 253, 242, 195,  64,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,  49, 238, 253, 253, 253, 253, 253, 253, 253, 253, 251,  93,  82,  82,  56,  39,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,  18, 219, 253, 253, 253, 253, 253, 198, 182, 247, 241,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,  80, 156, 107, 253, 253, 205,  11,   0,  43, 154,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,   0,  14,   1, 154, 253,  90,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0, 139, 253, 190,   2,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  11, 190, 253,  70,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  35, 241, 225, 160, 108,   1,   0,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  81, 240, 253, 253, 119,  25,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  45, 186, 253, 253, 150,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  16,  93, 252, 253,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0, 249, 253,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  46, 130, 183, 253, 253,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  39, 148, 229, 253, 253, 253, 250,
        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  24, 114, 221, 253, 253, 253, 253, 201,  78,
        0,   0,   0,   0,   0,   0,   0,   0,  23,  66, 213, 253, 253, 253, 253, 198,  81,   2,   0,
        0,   0,   0,   0,   0,   0,  18, 171, 219, 253, 253, 253, 253, 195,  80,   9,   0,   0,   0,
        0,   0,   0,   0,  55, 172, 226, 253, 253, 253, 253, 244, 133,  11,   0,   0,   0,   0,   0,
        0,   0,   0,   0, 136, 253, 253, 253, 212, 135, 132,  16,   0,   0,   0,   0,   0,   0,   0,
        ... ]
  }]
}
```

```
{
  "model_name": "mnist",
  "model_version": "1",
  "id": null,
  "parameters": null,
  "outputs": [{
    "name": "0",
    "shape": [1, 10],
    "datatype": "UINT8",
    "parameters": null,
    "data": [0, 0, 0, 2, 0, 254, 0, 0, 0, 0]
  }]
}
```

5.Dogfooding in FuriosaAI

5.1 MLPerf™ Benchmark Story

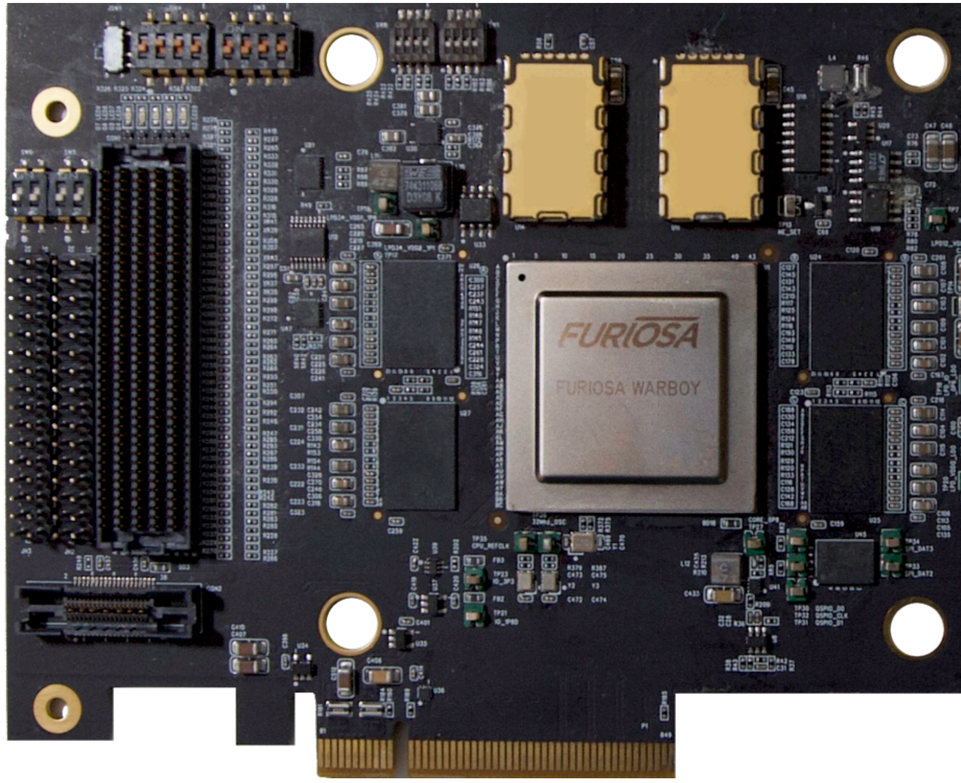
DEVIEW 2021

퓨리오사AI

전자신문 | 2021.09.23. | 네이버뉴스

퓨리오사AI, 글로벌 대회에서 세계적 기술 경쟁력 입증

인공지능(AI) 반도체 스타트업 **퓨리오사AI**가 첫 번째 실리콘 칩 '워보이'로 글로벌 AI 반도체 벤치마크 대회 'MLPerf' 추론 분야에서 뛰어난 성능을 달성하며 기술 경쟁력을 입증했다. MLP...

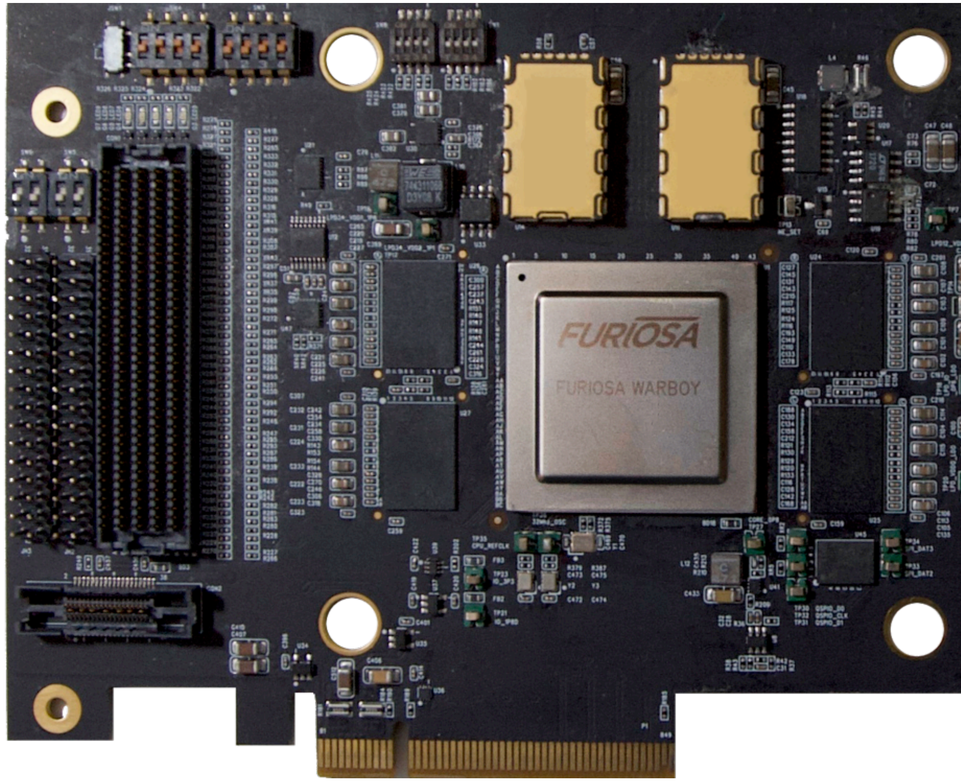


5.1 MLPerf™ Benchmark Story

DEVIEW 2021

급박하게 돌아갔던 지난 시간들…

- First engineering sample arrival: 7/19
- First End-to-End Test success: 7/23
- Benchmark Submission Due: 8/13



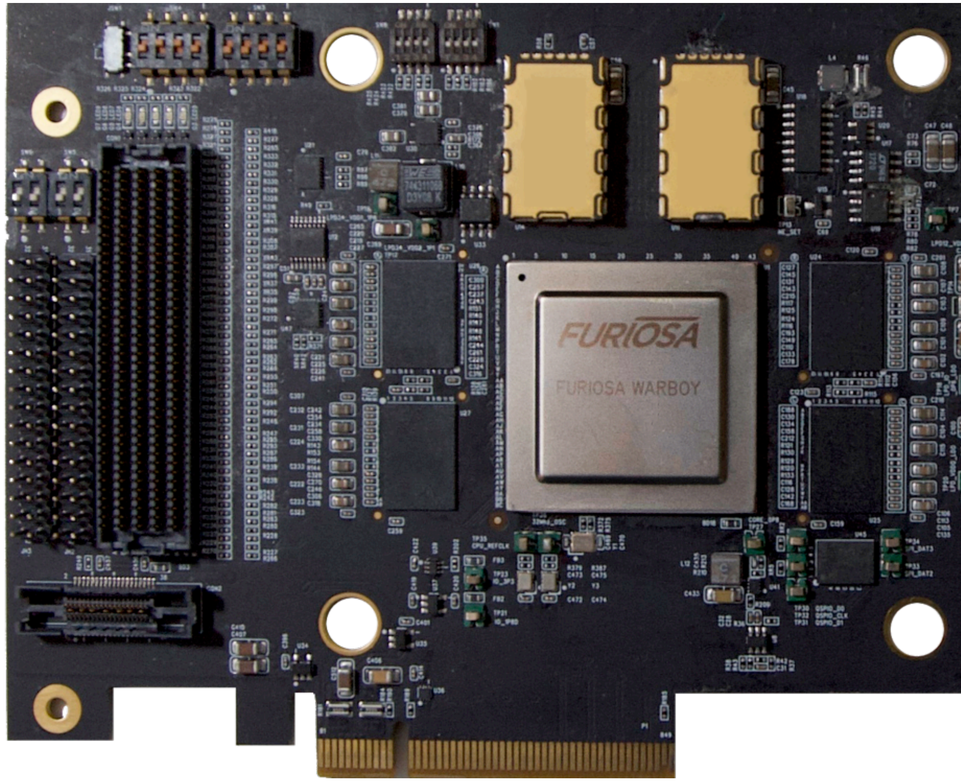
5.1 MLPerf™ Benchmark Story

DEVIEW 2021

NPU 개발 단계에서 NPU가 필요한 상황들

- 컴파일러 통합 테스트
- Inference Task 프로파일링
- SDK 지속적 통합 작업

- 이러한 작업들을 보장된 가용성 위에서 진행되도록!

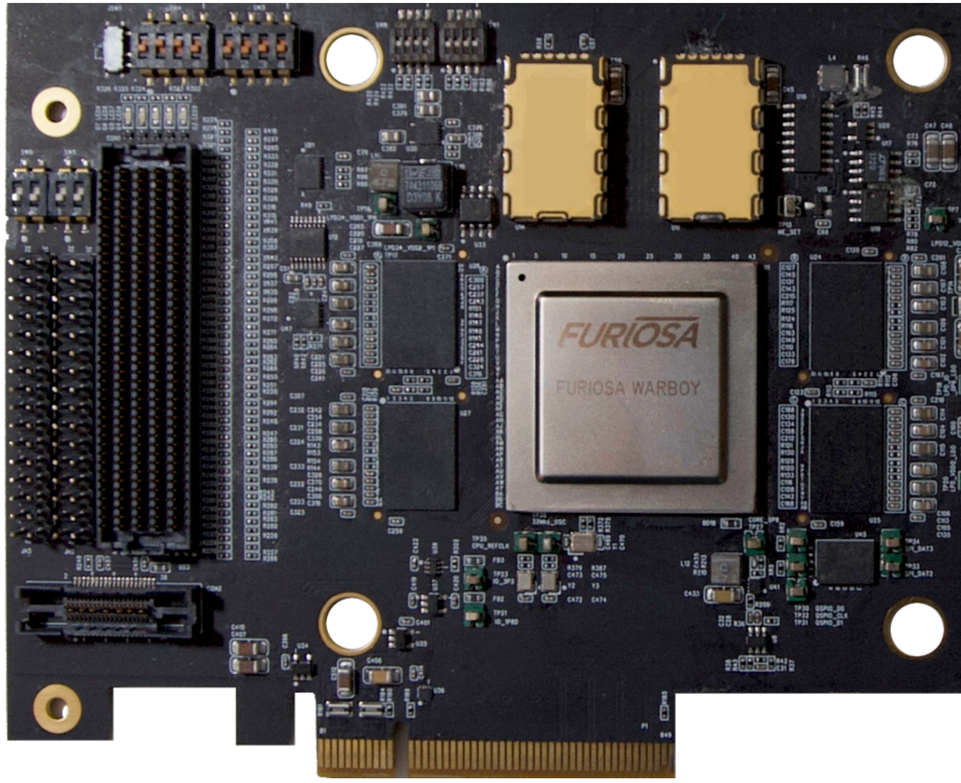


5.1 MLPerf™ Benchmark Story

DEVIEW 2021

Device Plugin으로 해결한 리소스 제공

- 코드 재사용으로 공급 시간 단축
- Pod을 만들어 개별 작업에 활용
  - 테스트
  - 프로파일링
- Jenkins를 통한 작업 자동화
  - 통합 테스트
  - 변경사항 별 벤치마크 성능 데이터 수집
- 모니터링을 통한 의도치 않은 자원 점유 점검

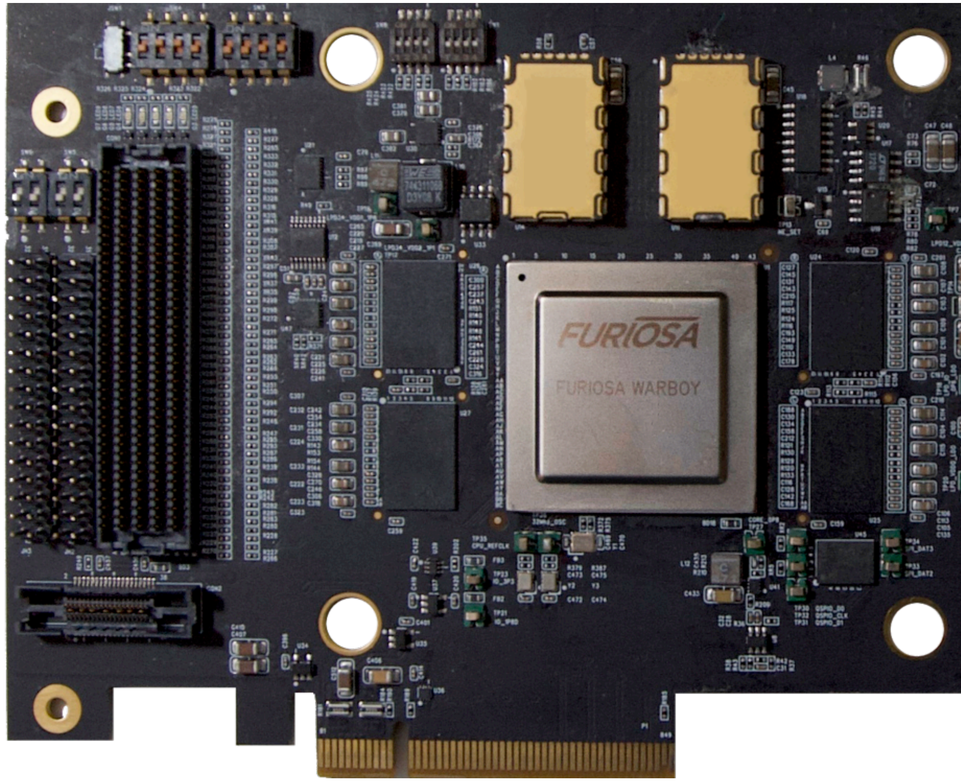


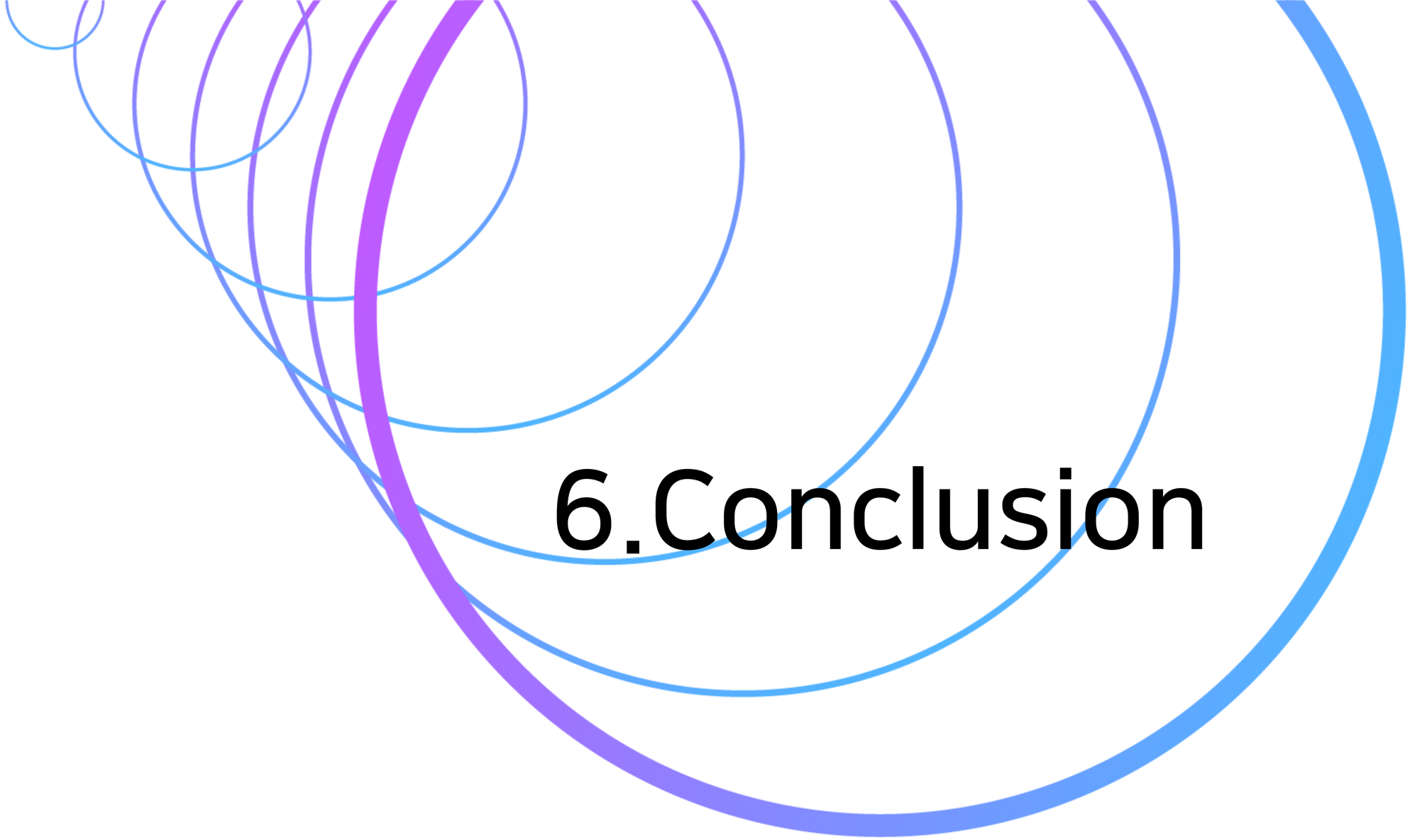
5.2 Post-mortem

DEVIEW 2021

Device Plugin 개선의 필요성

- 편의 기능 추가
  - configMap을 통한 설정 입력/변경
  - 할당 현황을 쉽게 확인할 수 있는 기능
- 디버깅
  - 이미 할당된 장비의 중복 할당 등
- 더 나은 k8s 통합을 위한 로드맵 계획 수립
  - 고도의 가상화
  - 고도의 추상화
  - write in Rust





6.Conclusion

6.1 효율적인 자원 활용

DEVIEW 2021

구슬이 서 말이라도 꿰어야 보배

- 값비싼 자원일수록 Utilization을 높이는 것이 중요!
- GPU 또는 NPU와 같은 장치들의 수량이 적절한지 확인
  - 사용량에 비해 너무 많거나, 너무 적지는 않은가?
- 모니터링도 k8s를 기반으로 수행할 수 있음

6.2 꼭 알아야 할까요?

DEVIEW 2021

Layer의 위치는 상대적인 개념

- k8s와 device plugin은,
  - Hardware와 Infrastructure 입장에서 바라보면 High Level,
  - 사용자와 ML Tool의 입장에서 바라보면 Low Level
- 대부분의 도구들은 이 기능을 기반에 두고 추상화 된 서비스를 제공하고 있음
  - 회사/팀 입맛에 맞는 도구를 직접 만들어야 한다면,
    이 기능을 유용하게 사용할 수 있음

6.3 훌륭한 HW를 만들기 위한 전제조건

DEVIEW 2021

Full-stack Product (Hardware to Software)

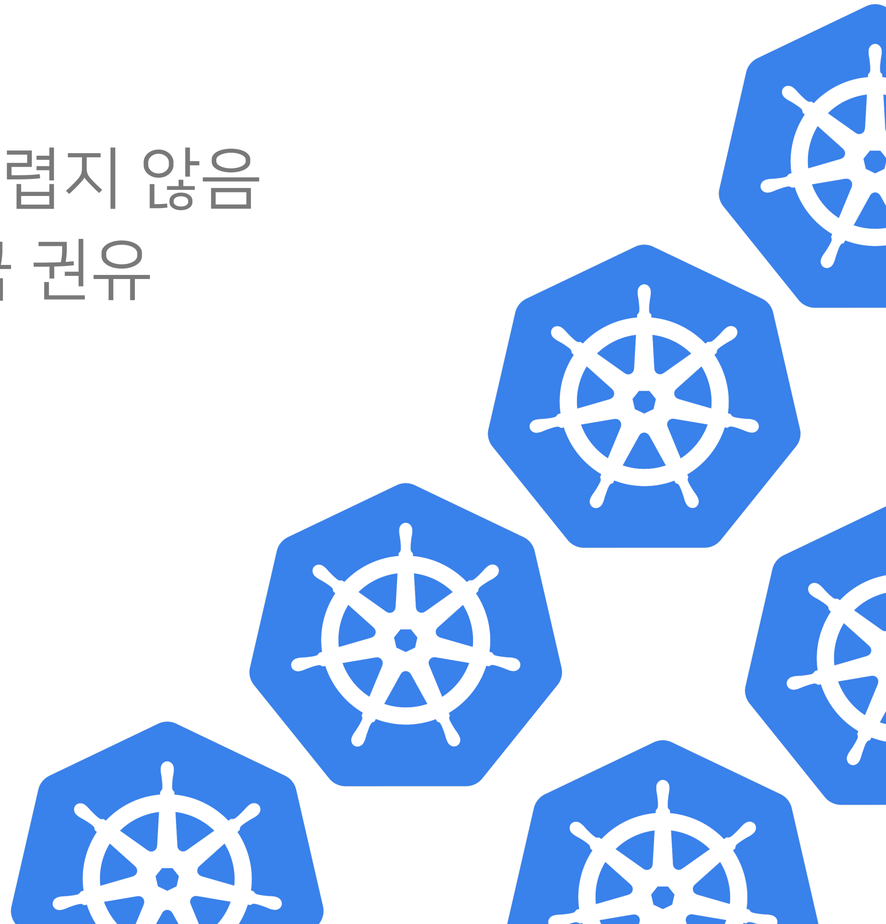
- 뛰어난 성능과 효율성을 자랑하는 하드웨어가 있더라도
  그 제품 자체만으로는 고객의 시선을 끌 수 없음
- Hardware 벤더들도 사용자 입장에서 편하게 사용할 수 있는
  Software와 서비스를 갖추고 있어야 함

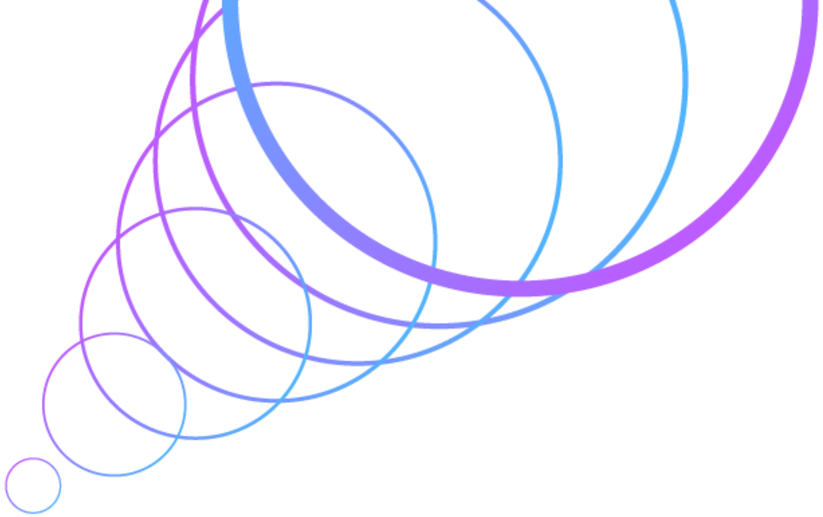
6.4 No Fear Kubernetes

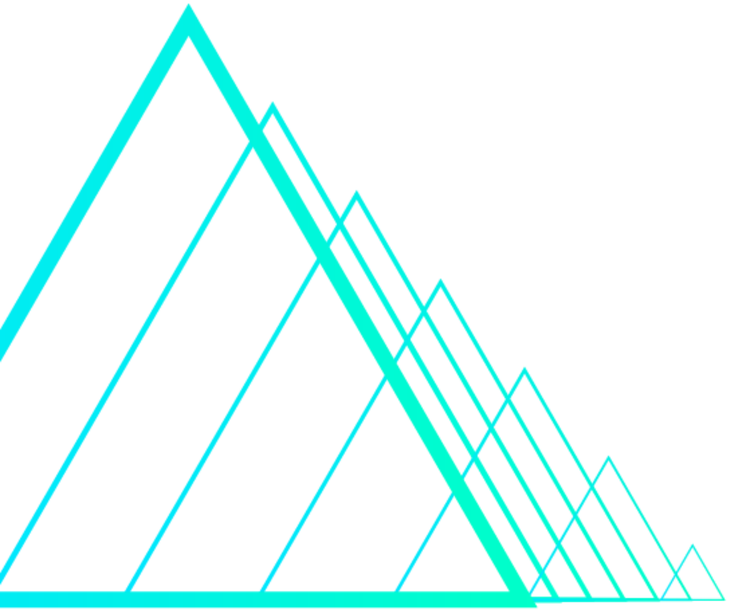
DEVIEW 2021

Kubernetes, 만만치 않다는데?

- k8s 운영과 관리는 어려운 것이 사실
- 연구/개발 목적으로 소규모로 사용하는 것은 생각만큼 어렵지 않음
- 참고할 수 있는 책, 글, 교육 자료 등이 많으니 도입을 적극 권유
- 기본적인 기능들만 사용해도 누릴 수 있는 효용이 많음







Q & A

Thank You