Dynamic Scaling of GPUs for Container Apps with Composable Disaggregated Infrastructure for AI Era

Fsas Technologies Inc.

Oct. 29th , 2024

Jin Hase, Zhang, Lei

Infrastructure requirements for AI era

Fsas Technologies

In AI era, it's important to use expensive GPUs as efficiently as possible

- Generative AI opens the age of AI
- Enormous computational resources are required

Conflicting requirements

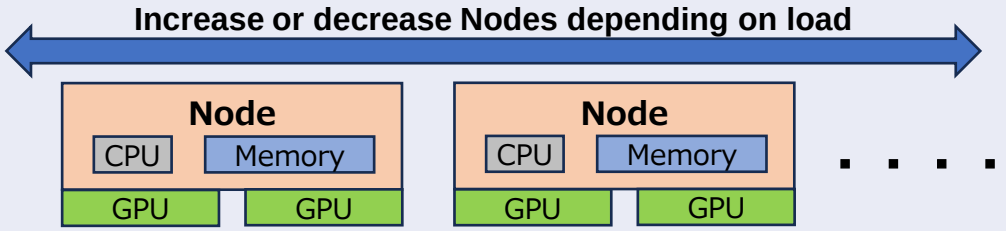
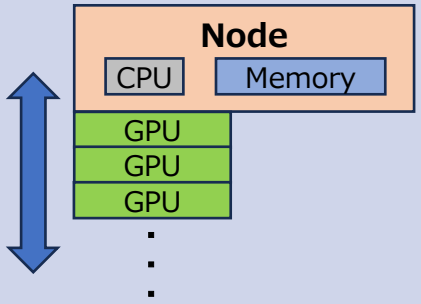
- Realizing sustainable world
- Reducing power consumption

- It is expected to balance high performance and power saving

Method for efficiently using GPU

Fsas Technologies

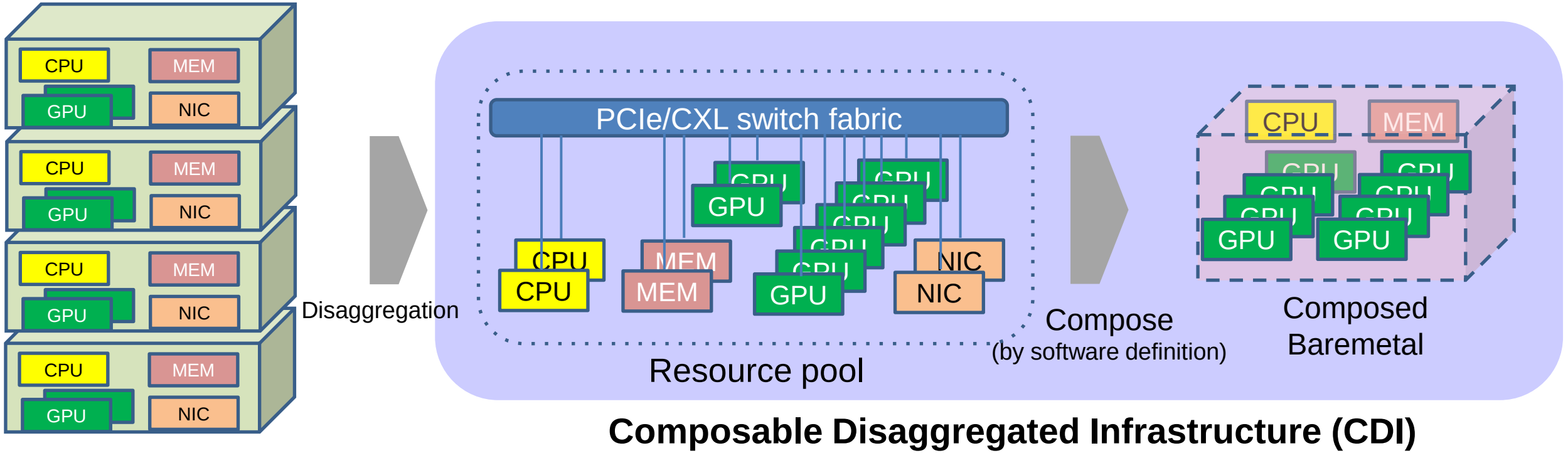
There are mainly three methods. We aim to realize the third method to achieve both high performance and power saving

	Methods	Performance	Power saving
1	Split GPUs(Time slicing, MPS, MIG, etc.)	× Limited number of GPUs	○ Efficient use of individual GPUs
2	Horizontal scaling of Nodes 	△ Adding Nodes takes time. This causes a temporary lack of performance	× If we only want a GPU, consuming unnecessary power for Nodes
3	Vertical scaling of devices Increase or decrease the number of GPUs attached to the Node depending on load 	○ Provide as many GPUs as needed quickly	○ No extra power because it provides only the GPU we need

Infrastructure for vertical scaling of devices

Fsas Technologies

Composable Disaggregated Infrastructure (CDI)
- Disaggregate servers into separate components as resource pool
- Create custom-made server by software definition (as we call Composed Baremetals)
- We can adjust the number of GPUs in each Composed Baremetal

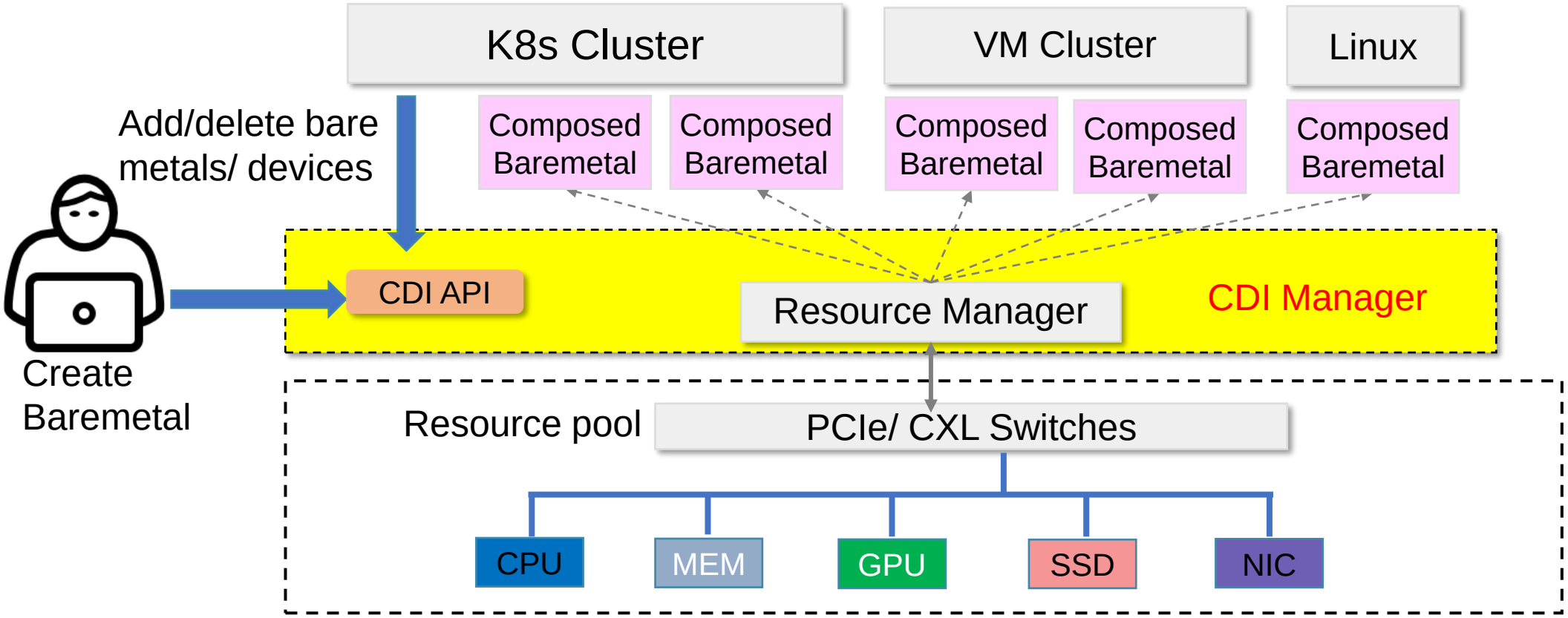


Composable Disaggregated Infrastructure (CDI)

CDI software stack

Fsas Technologies

Adjust the number of GPUs connected to Node in K8s cluster by CDI Manager
- Resource pool is achieved with PCIe/CXL switches
- CDI manager controls switch fabric to dynamically configure Composed Baremetals based on user demand



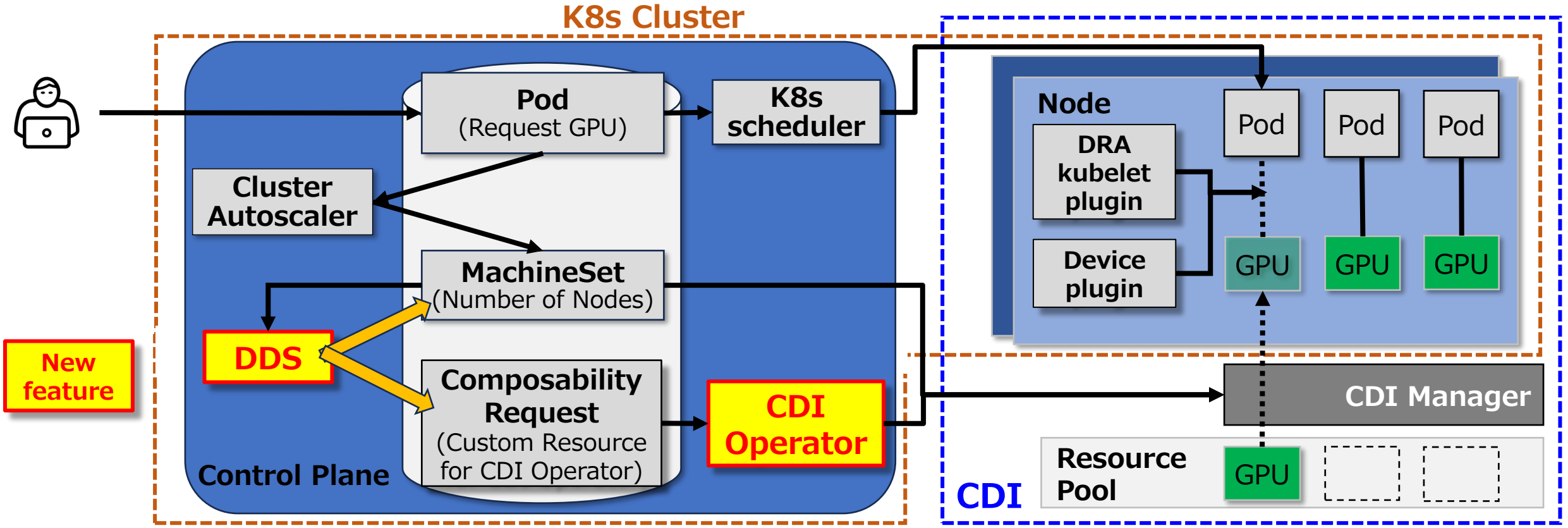
Dynamic Device Scaling by CDI

Fsas Technologies

Automatically attach/detach GPUs to K8s Nodes based on load

There are two key features:

- Dynamic Device Scaling (DDS): Determine whether to increase or decrease Node or GPU
- CDI Operator: Access CDI Manager and attach/detach GPUs as DDS decision

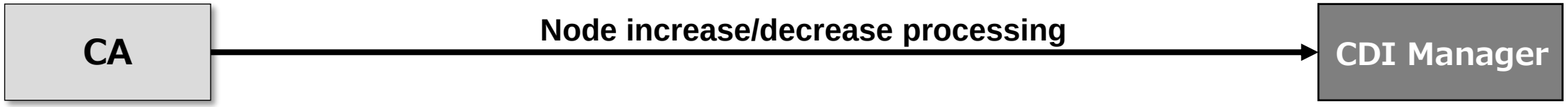


Basic Operation Flow

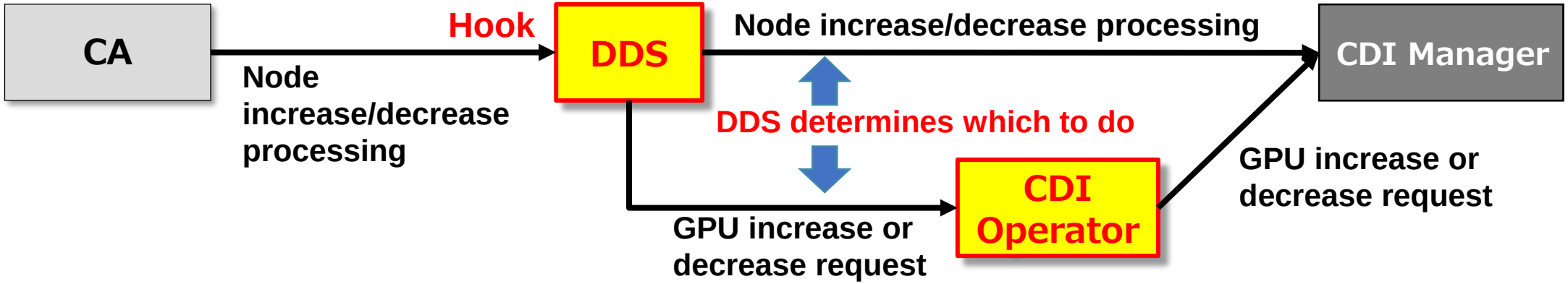
Fsas Technologies

- K8s has a feature called Cluster Autoscaler (CA) that horizontally scales Nodes
- DDS hooks up Node increase/decrease processing of CA
- DDS then determines whether to increase or decrease only GPUs or Nodes.

＜Current CA (Horizontal Node scaling) Processing＞



＜ Vertical device scaling Processing＞



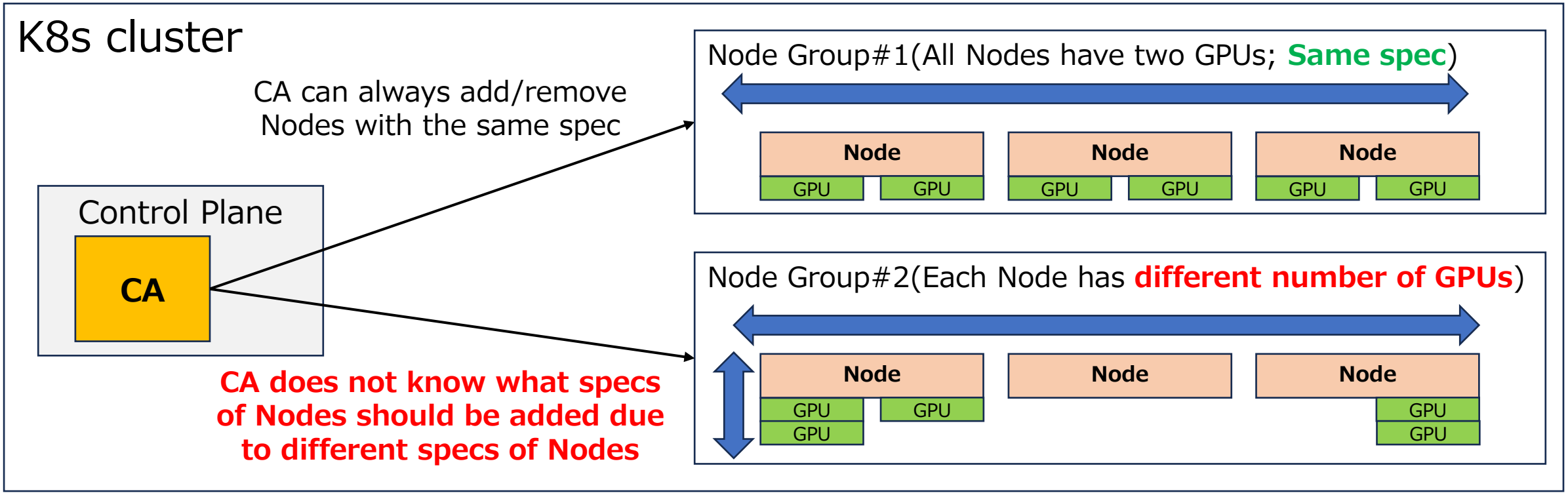
Challenges for achieving vertical scaling

Fsas Technologies

Need a mechanism to scale devices while keeping CA concepts
- CA groups Nodes with the same spec (*) and increase/decrease Nodes to the group
- Increasing / decreasing devices causes Nodes in the group to have different specs

→ This affects CA processing; CA does not know what specs of Node should be added

(*) Spec = CPU core, Memory size, number of GPUs, label, annotation and etc.



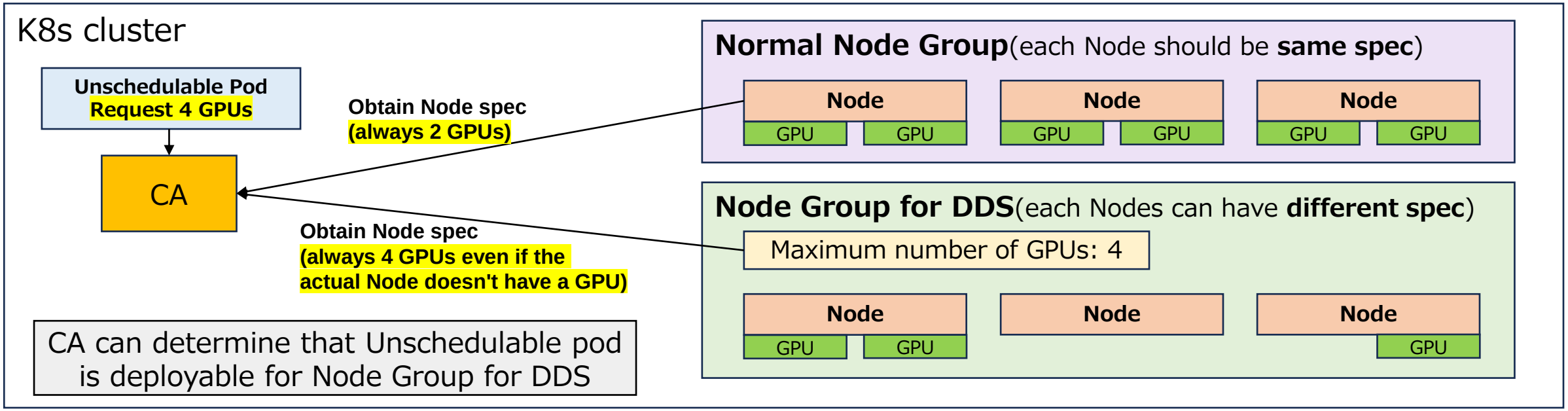
Our Solution:
Node Group for vertical scaling

Basic concept of Node Group for vertical scaling

Fsas Technologies

Show the maximum number of GPUs that can be attached to Node

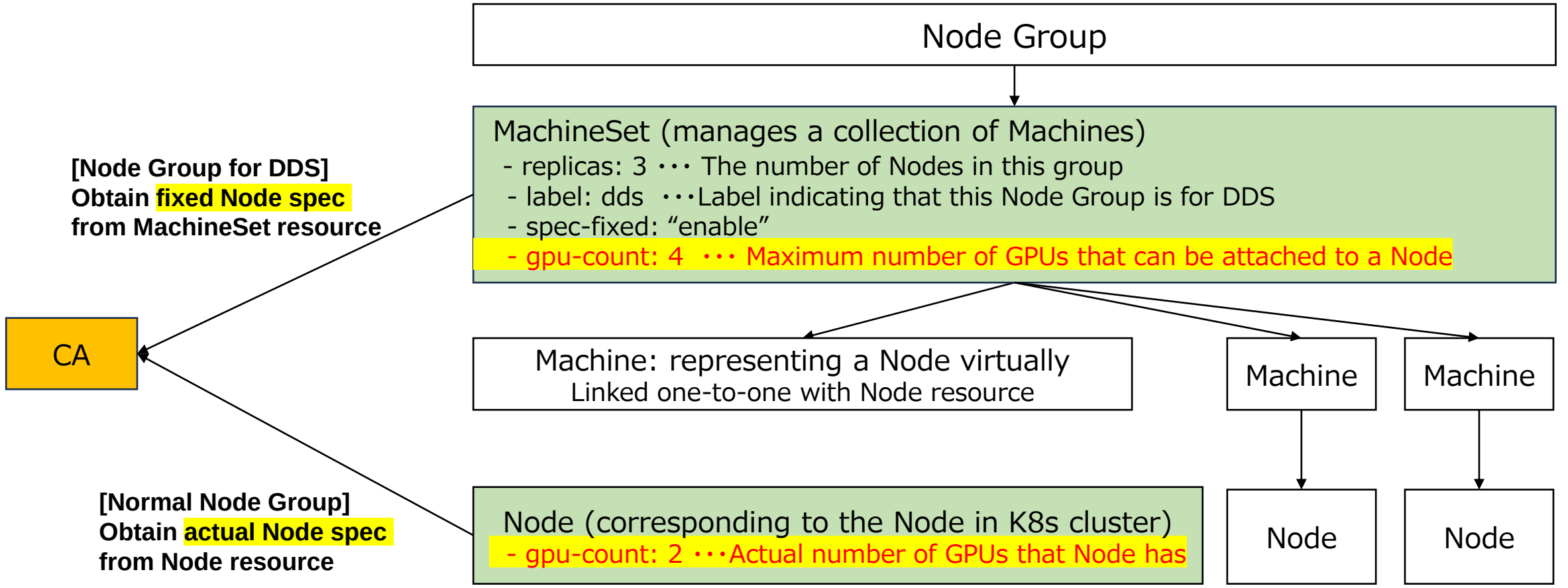
	Normal Node Group	Node Group for DDS
How to get Node spec	CA picks a random Node and check its specs to deploy Unschedulable Pod	Always show **fixed specs** to CA (Nodes in a Node Group can have different specs)
Value of Node spec	Actual Node spec (Actual attached GPUs, and same for each Node)	**Maximum number of GPUs** that can be attached to a Node (not the actual number of GPUs)



How to obtain Node spec (In case of using cluster-api)

Fsas Technologies

Include fixed spec value in the MachineSet resource. CA always gets the same spec (maximum number of GPUs) by referencing the spec from MachineSet

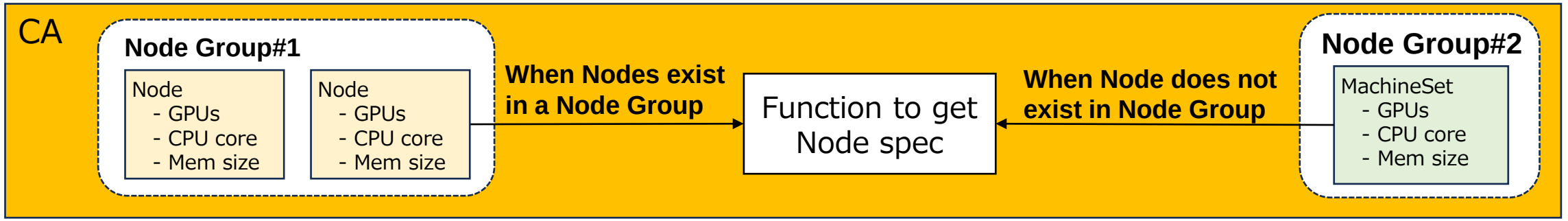


We are proposing a new option to MachineSet

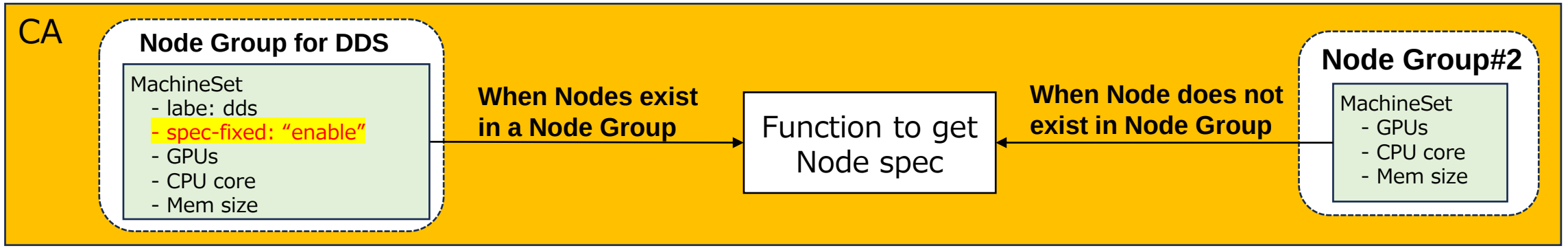
Fsas Technologies

- Current CA refers to MachineSet only if there are no Nodes in Node Group
- We add a new option called “spec-fixed” and make it always refer to a MachineSet when this option is enabled. This is a proposed feature for the K8s community

Current CA



Our proposal

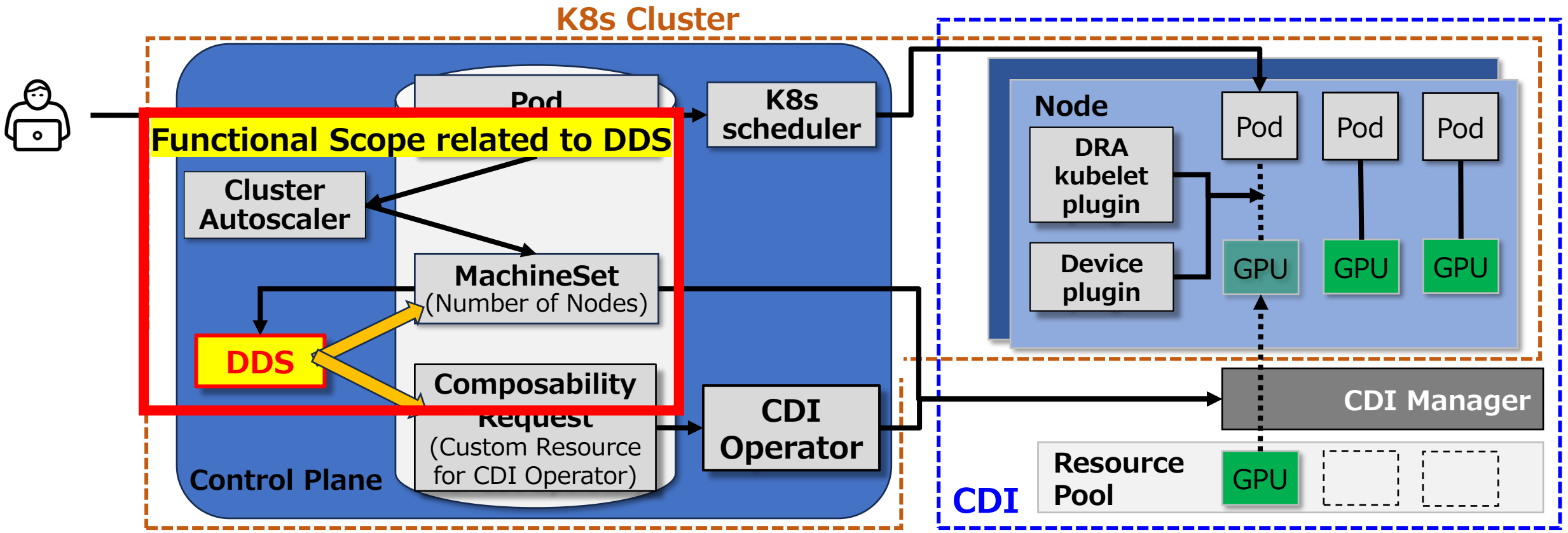


Dynamic Device Scaling

DDS Functional Scope

Fsas Technologies

- DDS keeps the existing CA concept.
- Therefore, DDS hooks CA processing (node increase/decrease) to determine if only GPU increase/decrease is required



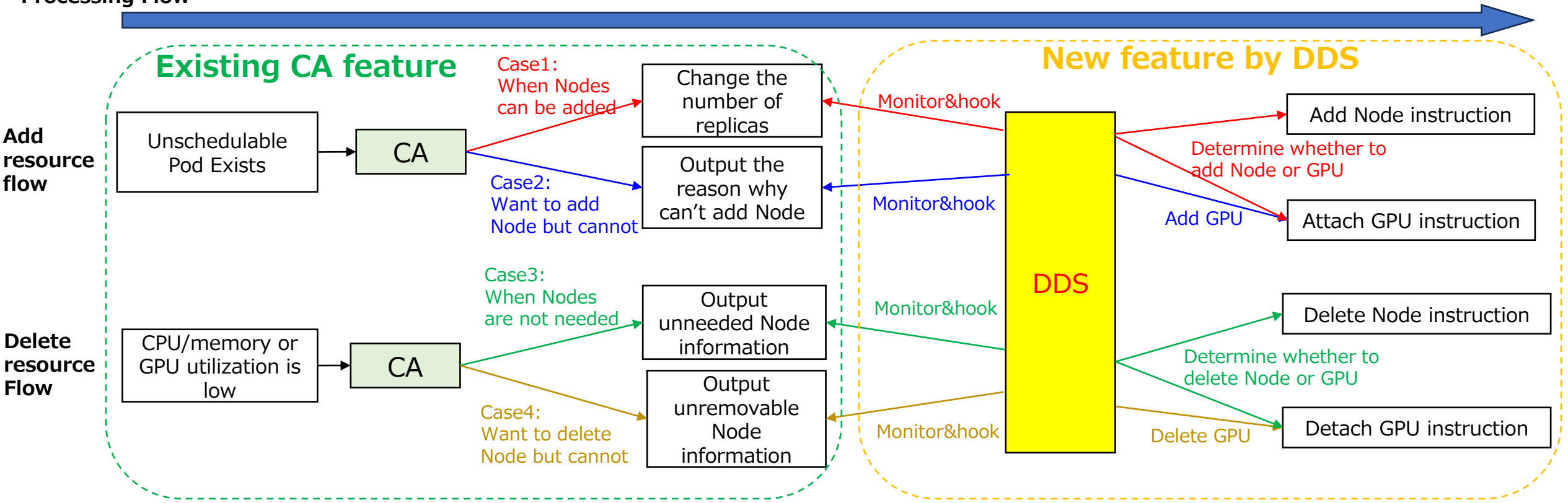
When DDS hooks CA processing and what it determines

Fsas Technologies

When CA makes Node add/delete decision, DDS intervenes to determine whether to add/remove GPUs or Nodes

Case1: When CA decides to add node →DDS determines whether adding only GPU or adding Node
Case2: When CA wants to add a node but cannot →DDS determines if it can add only GPU
Case3: When CA decides that the node is not needed →DDS determines whether deleting only GPU or Node
Case4: When CA wants to delete the node but cannot →DDS determines whether it can delete only GPU

Processing Flow

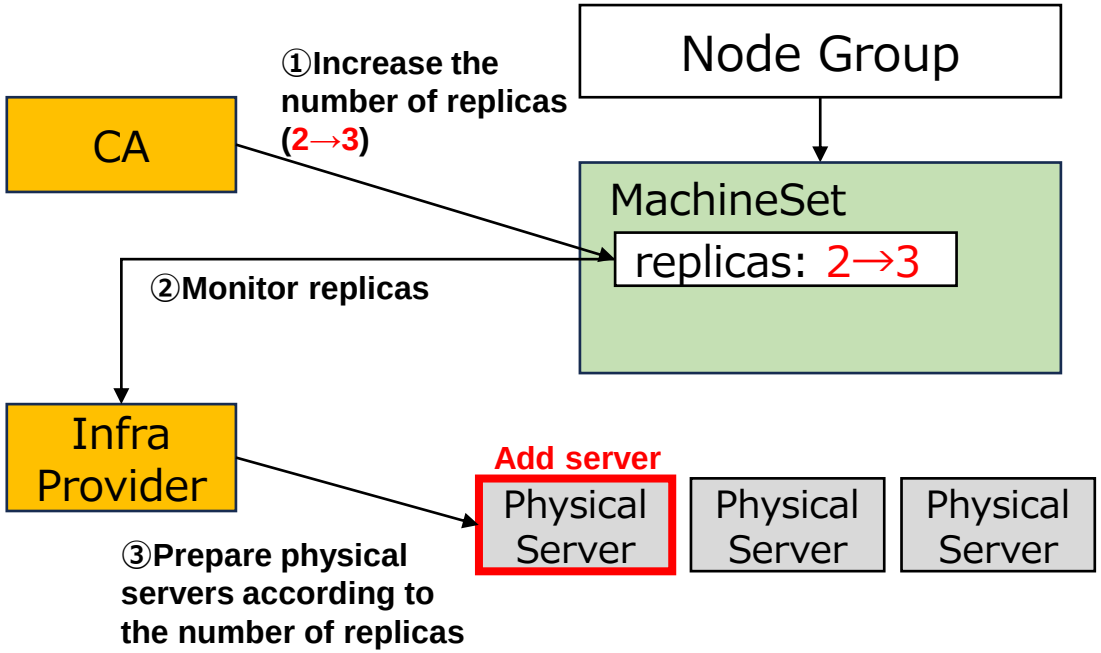


How to Hook CA Processing(Node increase case)

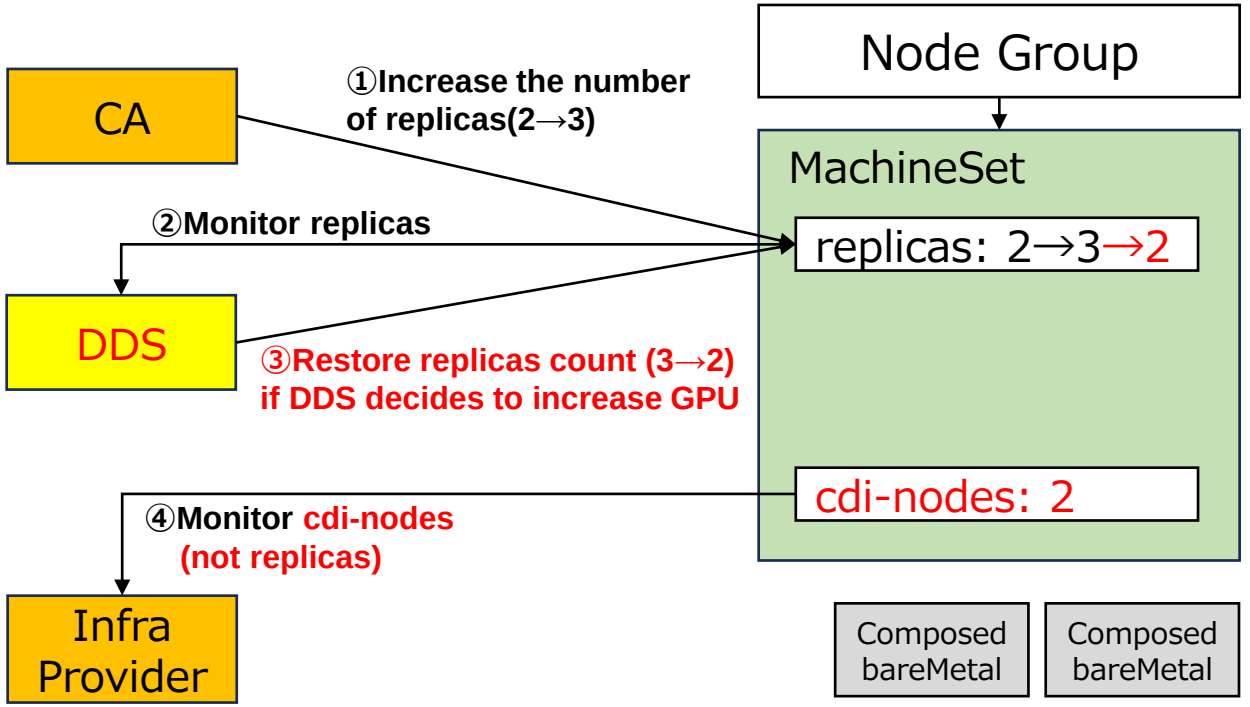
Fsas Technologies

- CA increases the number of replicas in MachineSet when adding more Nodes
- DDS monitors the increase in the number of “replicas” and goes into the process of determining whether to add GPU or Node

<Current CA process flow (Add Node)>



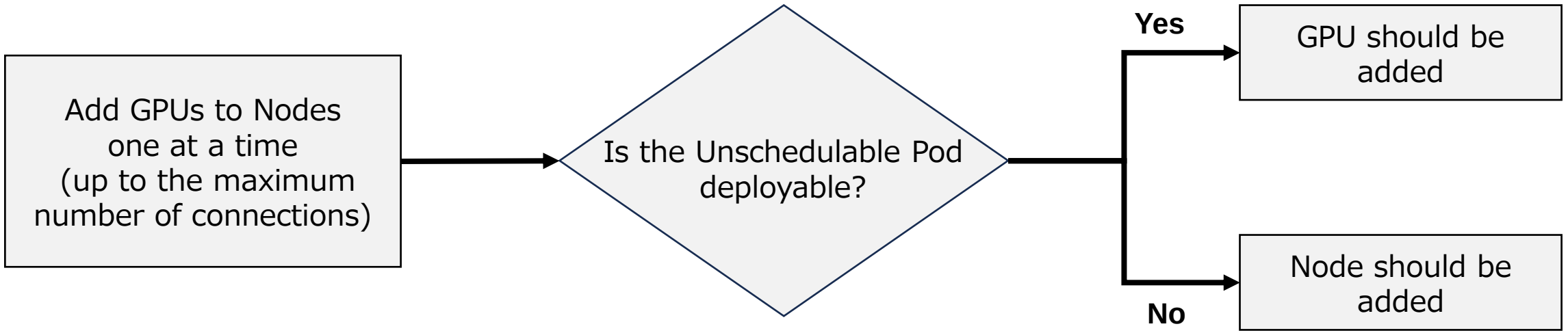
< DDS hooks up CA processing (Add GPU only)>



Decision logic for adding Nodes or GPUs

Fsas Technologies

Add GPUs to Nodes virtually in “Node Group for DDS” one at a time and calculate if an Unschedulable Pod can be deployed



Status of DDS proposals to the community

Fsas Technologies

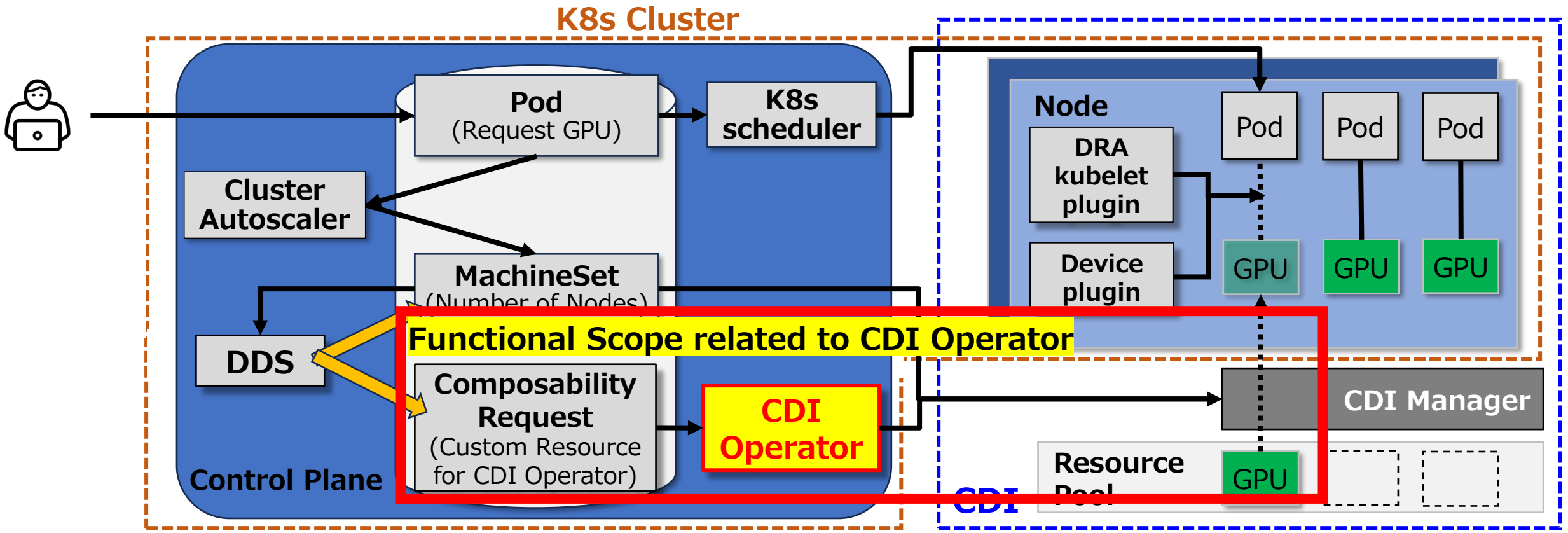
- Discussing node group for vertical scale
https://github.com/kubernetes/autoscaler/pull/7416
- Discussing the need for “spec-fixed” option with Cluster Autoscaler maintainers
https://github.com/kubernetes/autoscaler/issues/7380

CDI Operator

CDI Operator Functional Scope

Fsas Technologies

CDI Operator sends a request to CDI Manager to attach or detach GPUs

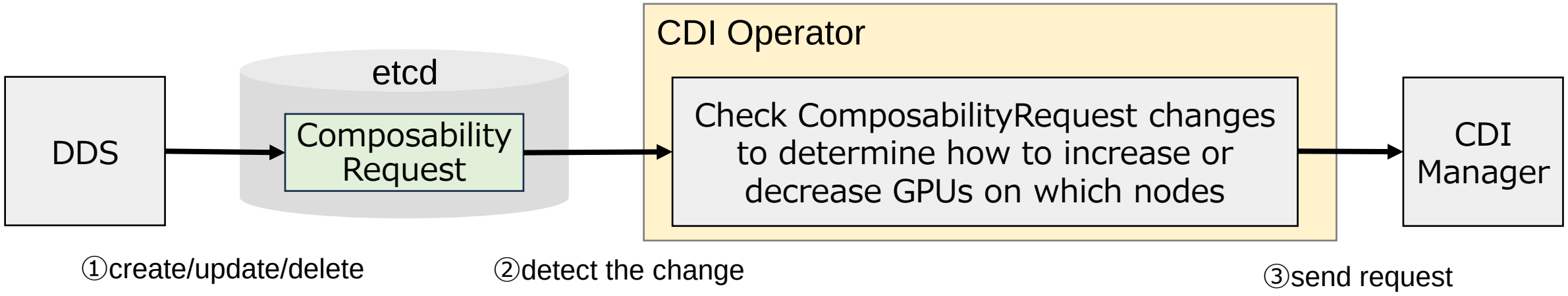


CDI Operator overview

Fsas Technologies

Processing flow

1: DDS creates/deletes/updates a custom resource for CDI Operator(ComposabilityRequest)
2: CDI Operator detects the change of ComposabilityRequest and then sends a request to CDI Manager to attach or detach GPUs



ComposabilityRequest data format

Fsas Technologies

We can specify which GPU model and how many are connected to which Node

Example for connecting 2 NVIDIA A100 GPUs to node1

```
apiVersion: v1alpha1
kind: ComposabilityRequest
metadata:
  name: composabilityrequest-sample
  namespace: cro-system
spec:
 resource:
   type: "gpu"
   size: 2
   model: "NVIDIA-A100-PCIE-40GB"
   target_node: "node1"
   force_detach: true
   allocation_policy: samenode
```

Specify Device Type
- Currently only GPU supported
- CXL memory and other device type will be supported in the future

The number of deices

Model name of device(optional)

Node to which you want to attach the device (optional)

Option to force detach even if GPU is used
(default is disabled)

Connection policy for GPU when no target node is specified
-samenode: Connect all devices to the same node as much as possible
-roundrobin: connect GPUs evenly to each node

CDI Operator Development Status

Fsas Technologies

- Discussing and developing in the following IBM Research's github
  They developed the base functionality and are now working together to improve it.
  https://github.com/IBM/composable-resource-operator

Demo

Demo Summary

Fsas Technologies

- Add GPU automatically
  1. A user create a Pod which request GPU, Because of insufficient GPU, the Pod is pending
  2. The DDS add GPU automatically
  3. The Pod is pending to go to 'Running' state.

- Add Node automatically
  1. A user create a Pod which request a lot of memory , Because of insufficient memory, the Pod is pending
  2. The DDS add node automatically
  3. The Pod is pending to go to 'Running' state.

Demo detail

Fsas Technologies

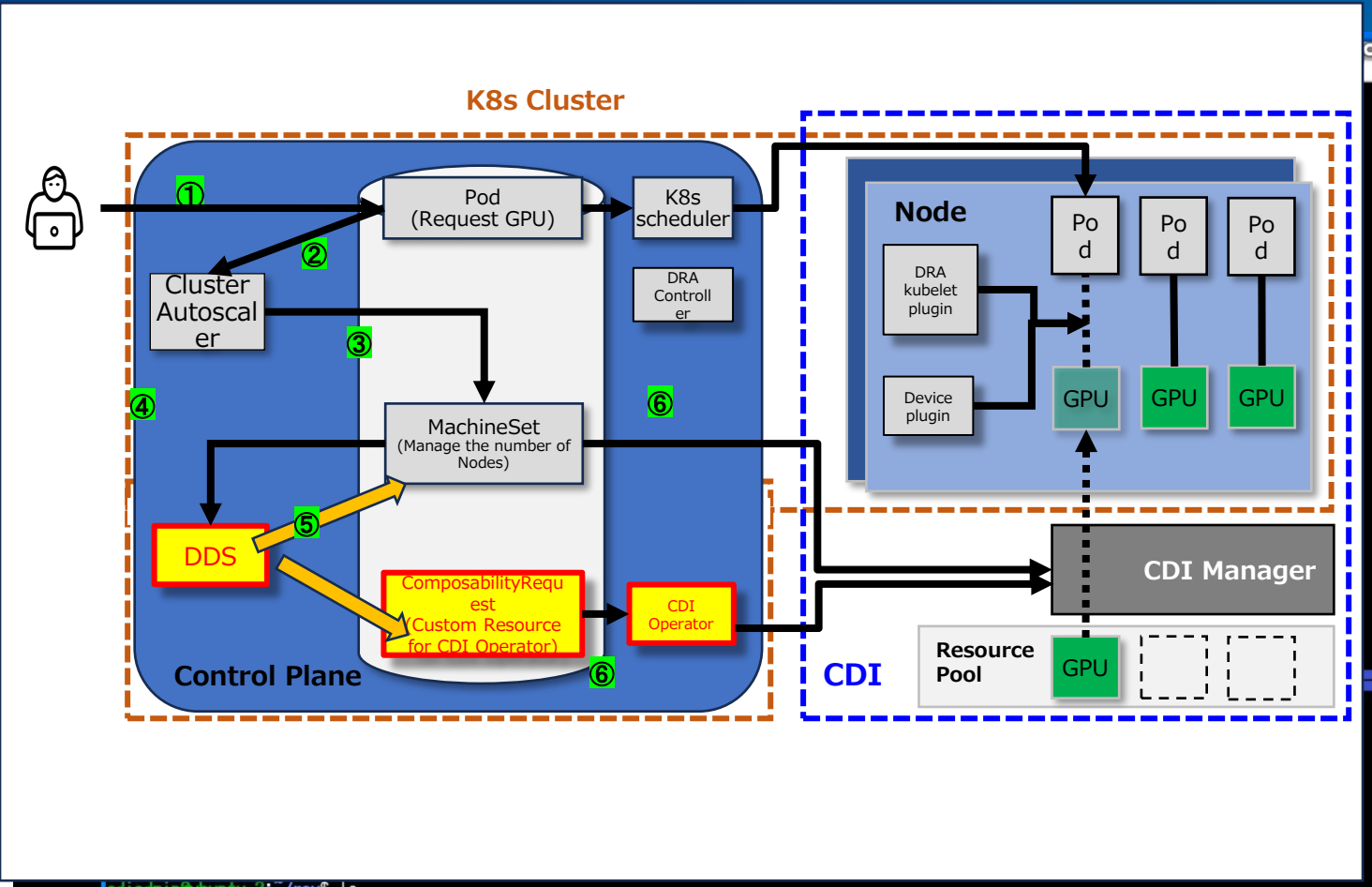
- Starting status
  - Worker Node1　(One GPU attached and in-use)
  - Worker Node2　(One GPU attached and in-use)

Create a pod(podA) which request GPU, Because of insufficient GPU, the pod is pending

- Add GPU automatically
  - Worker Node1　(Two GPUsattached) podA(pending→Running)
  - Worker Node2　(One GPU attached)

Create a pod(podB) which request a lot of memory, Because of insufficient memory, the pod is pending

- Add node automatically
  - Worker Node1　(Two GPUs attached)
  - Worker Node2　(One GPU attached)
  - Worker Node3　(no GPU attached) podB(pending→Running)



Monitor node/pod/GPU

```
Every 2.0s: echo node list  && oc get node|grep worker && echo &&echo  ...  ubuntu-2: Tue Oct 22 16:29:39 2

node list
worker-kcnae       Ready    worker                 4d4h    v1.29.8+f10c92d
worker-qeps3       Ready    worker                 7d22h   v1.29.8+f10c92d

pod list

GPU information per node.
Name:               worker-kcnae
  nvidia.com/gpu:     1
Name:               worker-qeps3
  nvidia.com/gpu:     1
```

```
cdiadmin@ubuntu-2:~$
```

Cdioperator log

```
cdiadmin@ubuntu-2:~/ray$ ls
clusterautoscaler.yaml         deploy-pod.yaml             sleep-gpu-pod.yaml   sleep-gpu-pod2.yaml  sleep_pod1.yaml  sleep_pod3.yaml
composabilityRequestcert.yaml  machinesetautoscaler.yaml  sleep-gpu-pod1.yaml  sleep_pod.yaml       sleep_pod2.yaml  test-gpu-pod.yaml
cdiadmin@ubuntu-2:~/ray$
```

Operation Windows

```
^C
cdiadmin@ubuntu-2:~$
```

DDS log

Future Outlook

Fsas Technologies

CA

- Discussing the detail specification for Node Group for DDS in SIG-Autoscaling community

DDS/CDI Operator

- Community standardization in parallel with development
- It's currently focused on GPU but will expand to other device types such as CXL memory

Support hot plug of devices for vertical scaling

- There are several features to use GPU in K8s environment.
  These features need to support hot plug
  - DynamicResourceAllocation (DRA)
  - GPU Operator

Fsas Technologies