



Accelerate your smart-city Edge AI deployments

With open source cloud-native infrastructure

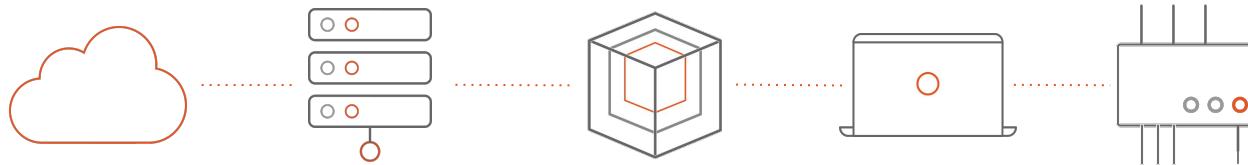
Gustavo Sanchez
Data & AI Solutions Architect
Canonical





Canonical's Ubuntu

From Cloud, Server, to Desktop and IOT



Bluefield DPU



NVIDIA H100 /
NVIDIA DGX

NVIDIA DGX
Superpods



Native integrations
with Ubuntu across
NVIDIA hardware



NVIDIA IGX



RTX

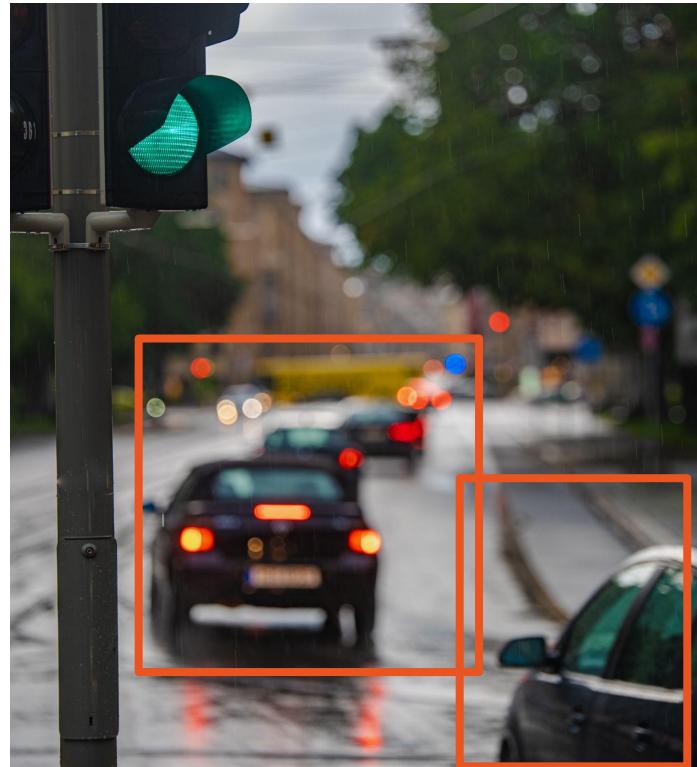


NVIDIA Orin



Smart Cameras

AI From Cloud to Edge





AI ML Developer Journey | Challenges

Cloud

Edge

1 - Infra

2 - Training

3 - Deploy

4 - Inference

5 - Operations

Cost Unpredictability

Complexity

Scalability

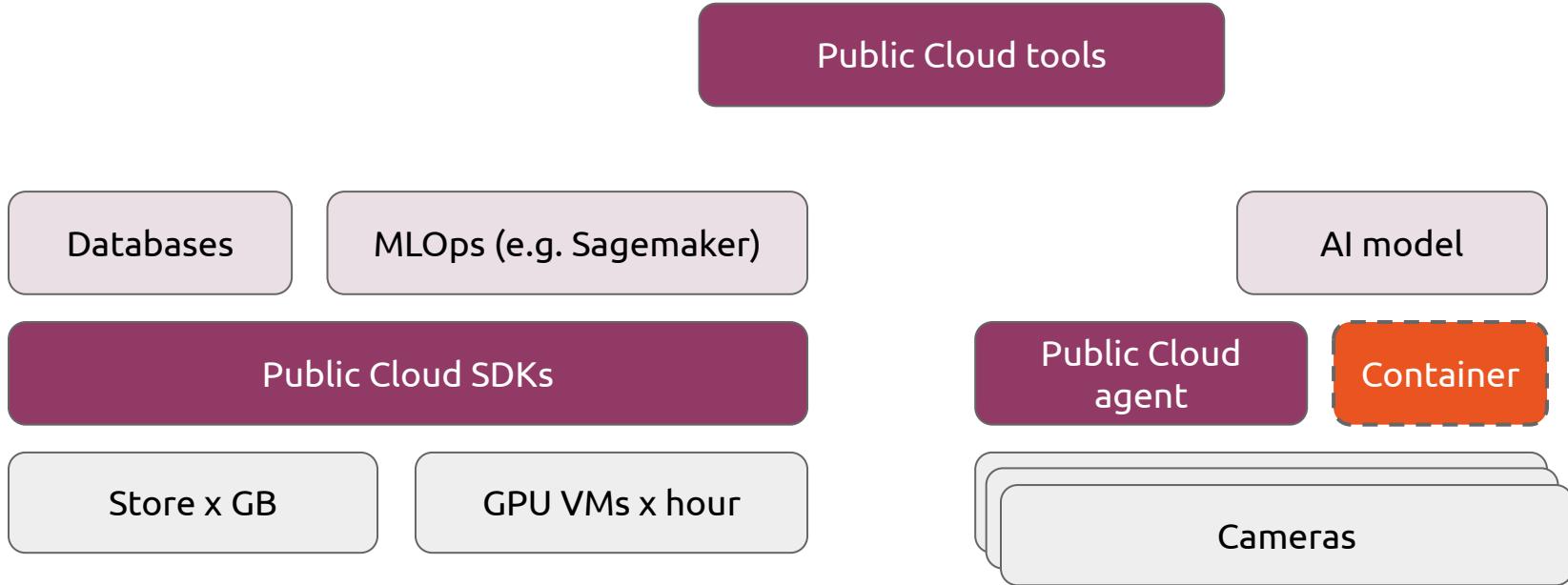
Security

Maintainability



Solution architecture

Baseline



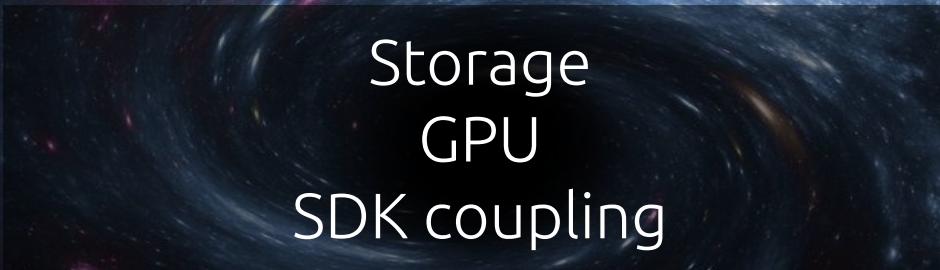


Cloud / Infrastructure

Extend your environment anywhere



Data Gravity



Storage
GPU
SDK coupling



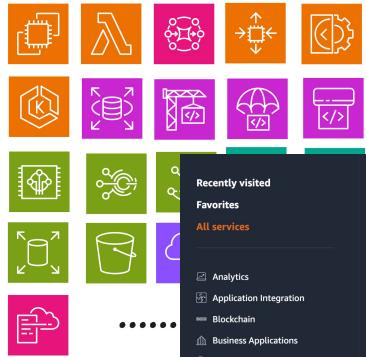
Cloud / Training

Reduce the complexity of your machine learning tools and operations



Training environment is **complex** to set up

A. Tons of clicks on GUI
or API calls from CLI



B. Complex IaC yaml files

```
resource "aws_subnet" "public_subnet" {
    vpc_id      = aws_vpc.vpc.id
    count       = var.subnet_count
    cidr_block  = cidrsubnet(var:cdr_vpc, var:cdr_network_bits, (count.index + length(split("", var:cdr_vpc.cidr_block))-1), 32)
    availability_zone = element(data.aws_availability_zones.all.names, count.index)
    map_public_ip_on_launch = true
}

tags = []
name = "public-${element(data.aws_availability_zones.all.names, count.index)}"

resource "aws_autoscaling_group" "my_sample_asg" {
    name           = var.asg_name
    launch_configuration = aws_launch_configuration.my_sample_lc.name // Reference form above
    min_size        = 2
    desired_capacity = 2
    max_size        = 4
    vpc_zone_identifier = aws_subnet.private_subnet.*.id
    health_check_type = "ELB"
    load_balancers   = [aws_elb.nginx_lb.name] // Add instances below Classic LB

    tag {
        key      = "Name"
        value    = "asg-nginx-test"
        propagate_at_launch = true
    }

    lifecycle {
        create_before_destroy = true
    }
}

resource "aws_nat_gateway" "nat_gateway_eip" {
    count      = 2
    vpc        = true
    depends_on = [aws_internet_gateway]
}

resource "aws_eip" "nat_gateway_eip" {
    count      = 2
    allocation_id = aws_eip.nat_gateway_eip.allocation_id
    subnet_id   = aws_subnet.public.id
    depends_on  = [aws_internet_gateway]
}

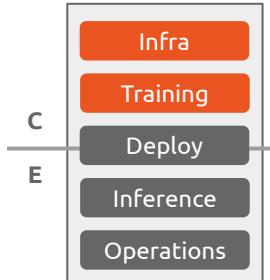
resource "aws_internet_gateway" "internet_gateway" {
    vpc_id      = aws_vpc.vpc.id
    depends_on  = [aws_eip.nat_gateway_eip]
}

resource "aws_sg" "lb_sg" {
    name           = "${var.sg_name}-lb"
    description    = "Managed by Terraform"
    vpc_id         = aws_vpc.vpc.id
    egress {
        from_port  = 0
        to_port    = 0
        protocol   = "-1"
        cidr_blocks = ["0.0.0.0/0"]
    }
}
```



Solution architecture

Open-source (cloud)



Public Cloud tools

Databases

MLOps (with Canonical Kubeflow)

Kubernetes (with Canonical MicroK8s)

Any laptop, VM, or server

Training

Anywhere

Public Cloud
agent

Container

Cameras

Inference
Edge



Simplify your training environment - in 3 steps

3 Steps

- 1 snap install microk8s
- 2 microk8s enable gpu
- 3 juju deploy kubeflow



1. Set up a cloud agnostic environment

In one line

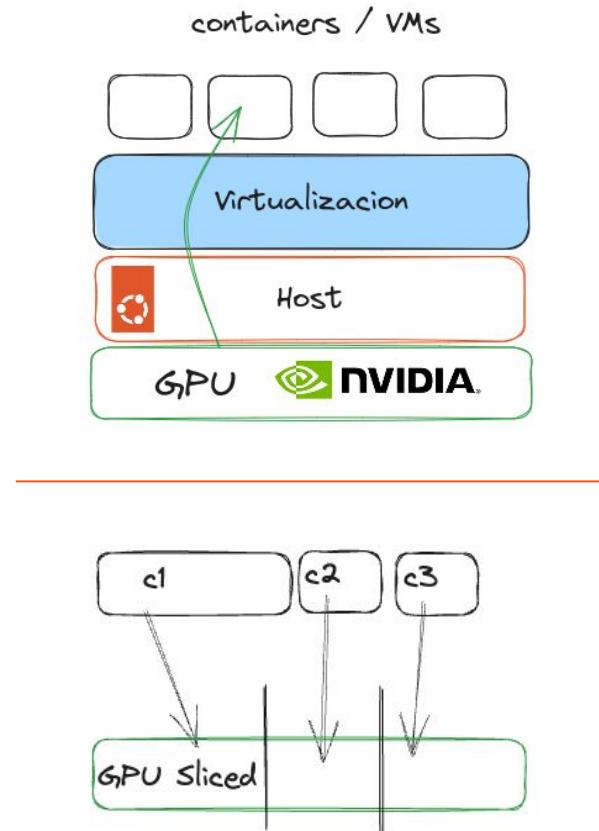
```
ubuntu@blue:~$ sudo snap install microk8s --channel 1.28/stable --classic  
microk8s (1.28/stable) v1.28.7 from Canonical✓ installed  
ubuntu@blue:~$ █
```





2. GPU Drivers and Scheduling In one line

```
ubuntu@blue:~$ sudo microk8s enable gpu
Infer repository core for addon gpu
Enabling NVIDIA GPU
Addon core/dns is already enabled
Addon core/helm3 is already enabled
Checking if NVIDIA driver is already installed
GPU 0: NVIDIA GeForce GTX 1660 Ti (UUID: GPU-15d01162-
Using host GPU driver
"nvidia" has been added to your repositories
Hang tight while we grab the latest from your chart re...
...Successfully got an update from the "nvidia" chart
Update Complete. *Happy Helming!*
NAME: gpu-operator
LAST DEPLOYED: Mon Mar 11 03:23:35 2024
NAMESPACE: gpu-operator-resources
STATUS: deployed
REVISION: 1
TEST SUITE: None
NVIDIA is enabled
ubuntu@blue:~$
```





3. MLOps E2E environment In two lines

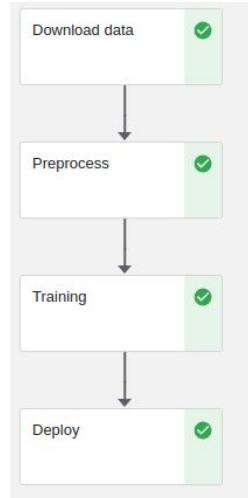
```
ubuntu@blue:~$ juju bootstrap microk8s
```

```
ubuntu@blue:~$ juju deploy kubeflow --channel 1.7/stable --trust
```

Experimenting



Training Pipelines



Model Versioning



Model Serving





Cloud to Edge / Deploy

Scale your solution with ease across your fleet of devices



Deploying a model is a multi-step process

```
1 # ===== 0. Register
2 camera_1 $ sudo apt install default-jre
3 camera_1 $ export AWS_ACCESS_KEY_ID=<AWS_ACCESS_KEY_ID>
4 camera_1 $ export AWS_SECRET_ACCESS_KEY=<AWS_SECRET_ACCESS_KEY>
5 camera_1 $ export AWS_SESSION_TOKEN=<AWS_SESSION_TOKEN>
6 camera_1 $ curl -s https://d2s8p88vqu9w66.cloudfront.net/releases/greengrass-nucleus-latest.zip > greengrass-
    nucleus-latest.zip && unzip greengrass-nucleus-latest.zip -d GreengrassInstaller
7 camera_1 $ sudo -E java -Droot="/greengrass/v2" -Dlog.store=FILE -jar ./GreengrassInstaller/lib/Greengrass.jar -
    -aws-region us-east-2 --thing-name GreengrassQuickStartCore-18e2ce04721 --thing-group-name
    GreengrassQuickStartGroup --component-default-user ggc_user:ggc_group --provision true --setup-system-service
    true --deploy-dev-tools true
8 [...]
9 camera_999 $ [...]
10
11 # ===== 1. Publish
12 develop $ push repository/my_app:1.0.1
13
14 # ===== 2. Rollout
15 camera_1 $ pull repository/my_app:1.0.1
16 camera_1 $ rollout 1.0.1
17 [...]
18 camera_999 $ [...]
19 # [breaks]
20
21 # ===== 3. Rollback
22 camera_1 $ rollout 1.0.0
23 camera_2 $ rollout 1.0.0
24 [...]
25 camera_999 $ [...]
```



Deploying a model needs to be scalable

One line publish

```
develop $ upload --release=1.0/stable my_app
```

Transactional updates in case of failure





Edge / Inference

Security outside your datacenter



Traditional access to hardware

A. Unsecure full access

```
1 docker run \
2   --privileged \
3   --ipc=host \
4   --device=/dev/video0 \
5   -v <host-path>:<container-path>
```

B. Complex granular security rules

```
1 # Description: This interface allows for setting CPU tunables
2 /sys/devices/system/cpu/*w, r,
3 /sys/devices/system/cpu/cpu*/online rw,
4 /sys/devices/system/cpu/smt/s,
5 /sys/devices/system/cpu/smt/control w,
6
7 /sys/module/cpu_boost/parameters/input_boost_ms w,
8 /sys/module/cpu_boost/parameters/input_boost_ns rw,
9 /sys/devices/system/cpu/cpu[0-3]/cpuFreq/scaling_min_freq w,
10 /sys/devices/system/cpu/cpu[0-3]/cpuFreq/scaling_max_freq w,
11 /sys/devices/system/cpu/cpu[0-3]/cpuFreq/ct/lat_ns comx w,
12 /sys/devices/system/cpu/cpu[0-3]/core_ct/busy_up_thres rw,
13 /sys/devices/system/cpu/cpu[0-3]/core_ct/busy_down_thres rw,
14 /sys/devices/system/cpu/cpu[0-3]/core_ct/busy_low_thres rw,
15 /sys/devices/system/cpu/cpu[0-3]/core_ct/busy_high_thres rw,
16 /sys/devices/system/cpu/cpu[0-3]/core_ct/nr_area_assist_thresh rw,
17 /sys/devices/system/cpu/cpu[0-3]/core_ct/enable rw
18
19 # https://www.kernel.org/doc/html/latest/admin-guide/pm/cpufreq.html#policy-interface-in-sysfs
20 /sys/devices/system/cpu/cpufreq/policy/energy_performance_preference u,
21 /sys/devices/system/cpu/cpufreq/policy/scaling_governor u,
22 /sys/devices/system/cpu/cpufreq/policy/scaling_max_freq w,
23 /sys/devices/system/cpu/cpufreq/policy/scaling_min_freq w,
24 /sys/devices/system/cpu/cpufreq/policy/scaling_setpoint u,
25 /sys/devices/system/cpu/cpufreq/policy/scaling_update_rate ns,
26 /sys/devices/system/cpu/cpufreq/policy/schedutil/hispeed_limit_us w,
27 /sys/devices/system/cpu/cpufreq/policy/schedutil/hispeed_freq rw,
28 /sys/devices/system/cpu/cpufreq/policy/schedutil/pl1_rw,
29 /sys/devices/system/cpu/cpufreq/policy/pl2_rw
30
31 # https://www.kernel.org/doc/html/latest/admin-guide/pm/intel_pstate.html#user-space-interface-in-sysfs
32 /sys/devices/system/cpu/intel_pstate/locked_state/rw,
33 /sys/devices/system/cpu/intel_pstate/mng_dynamic_boost w,
34 /sys/devices/system/cpu/intel_pstate/max_perf_pct w,
35 /sys/devices/system/cpu/intel_pstate/min_perf_pct w,
36 /sys/devices/system/cpu/intel_pstate/no_turbo w,
37 /sys/devices/system/cpu/intel_pstate/status w,
38
39 /proc/sys/kernel/sched_ugrade_nr,
40 /proc/sys/kernel/sched_downmigrate_nr,
41 /proc/sys/kernel/sched_group_unicrate_nr,
42 /proc/sys/kernel/sched_group_migrateto_nr,
43 /proc/sys/kernel/sched_walt_rotate_big_tasks rw,
44 /proc/sys/kernel/sched_burst_nr,
45
46 # see https://www.osadl.org/monitoring/add-on-patches/4.16.7-r1...4.16.15-r7/sched-add-per-cpu-load-measurement.patch.html
47 /proc/driver/rtuntime@0x1,cpu[0-3]@/data r,
48 /proc/driver/rtuntime@0x1,cpu[0-3]@/reset v,
```



But.. is that how your phone works?

Secure and simple IoT



Only 2 lines

```
1  webcam-daemon:  
2    command: bin/python $SNAP/webcam.py  
3    daemon: simple  
4    plugs:  
5      - camera  
6      - opengl
```

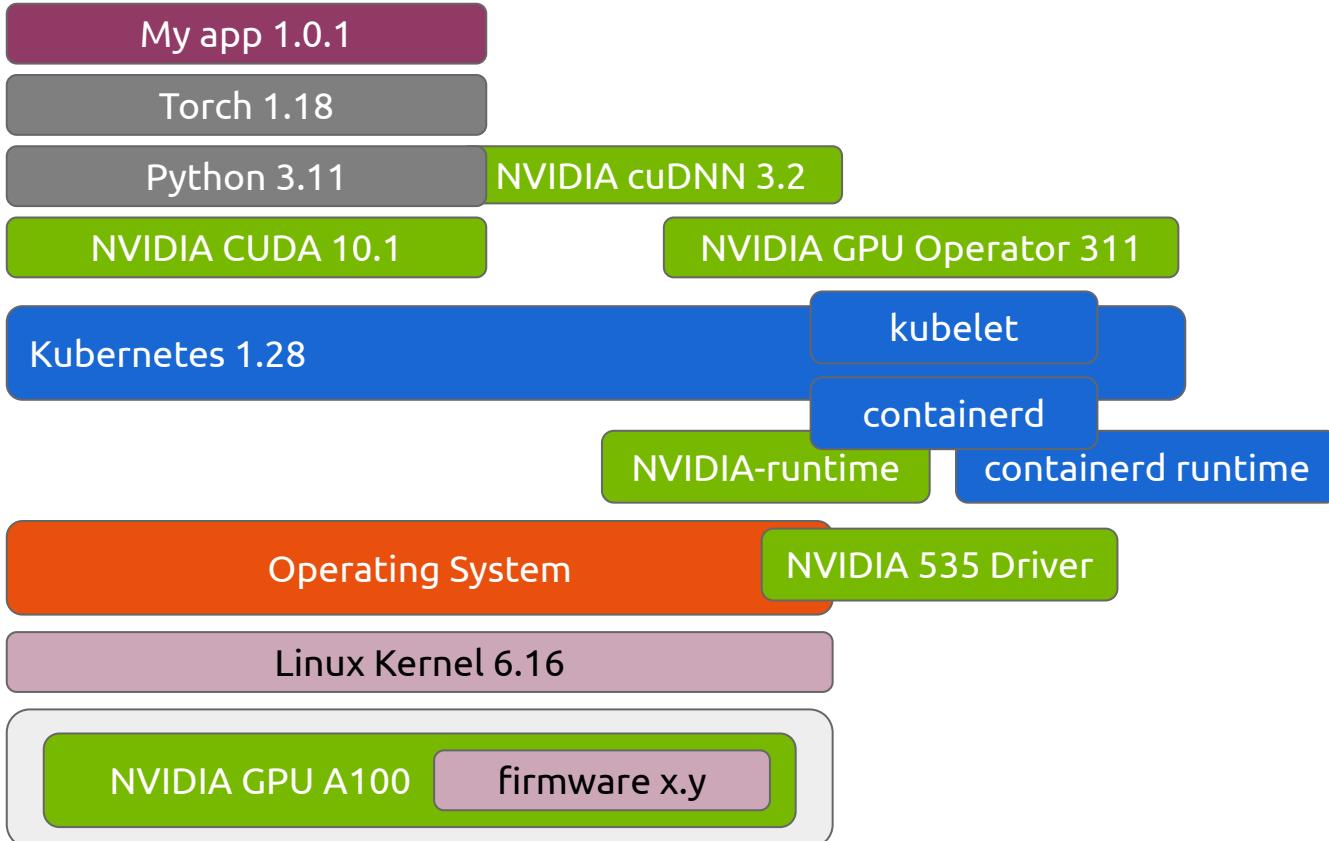


Edge / Operations

Robustness out in the wild

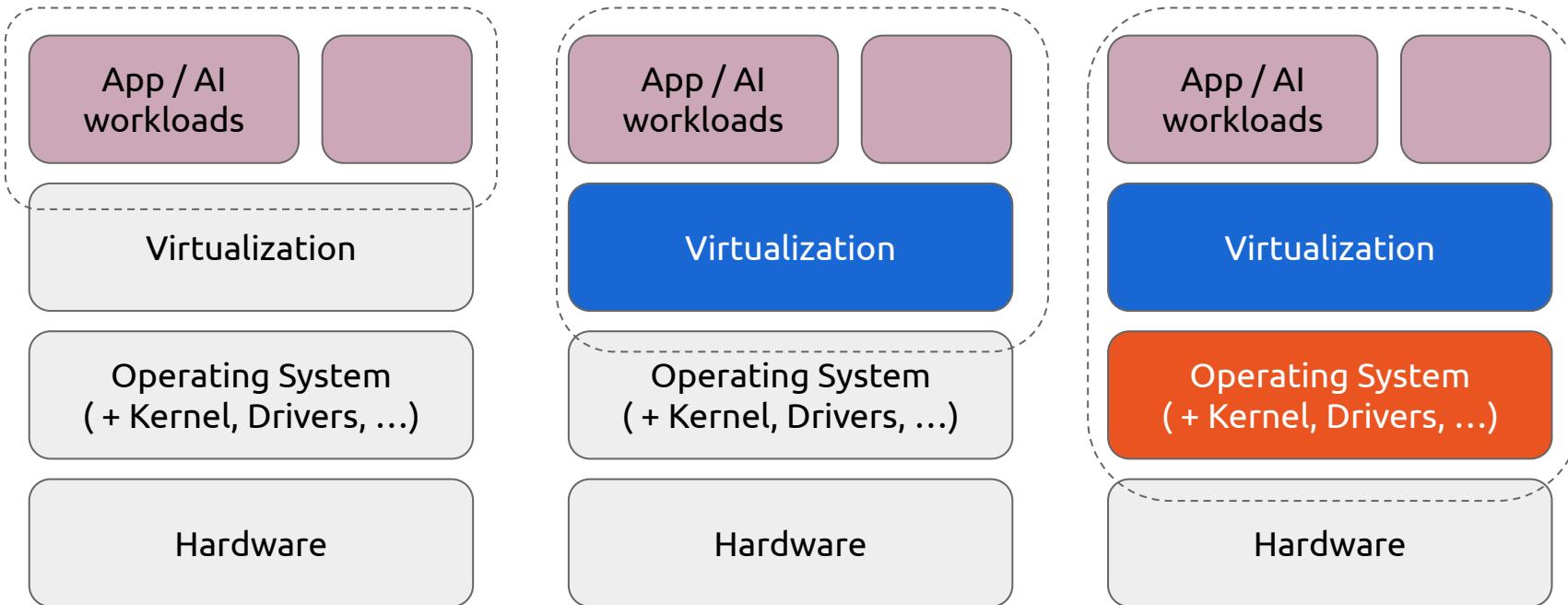


The AI Stack: more than just the model





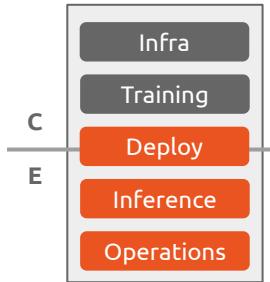
Updates and patches across all layers





Solution architecture

Open-source (edge)



Snapcraft

Databases

MLOps (with Kubeflow)

Kubernetes (with MicroK8s)

Any laptop, VM, or server

Training

Anywhere

AI model

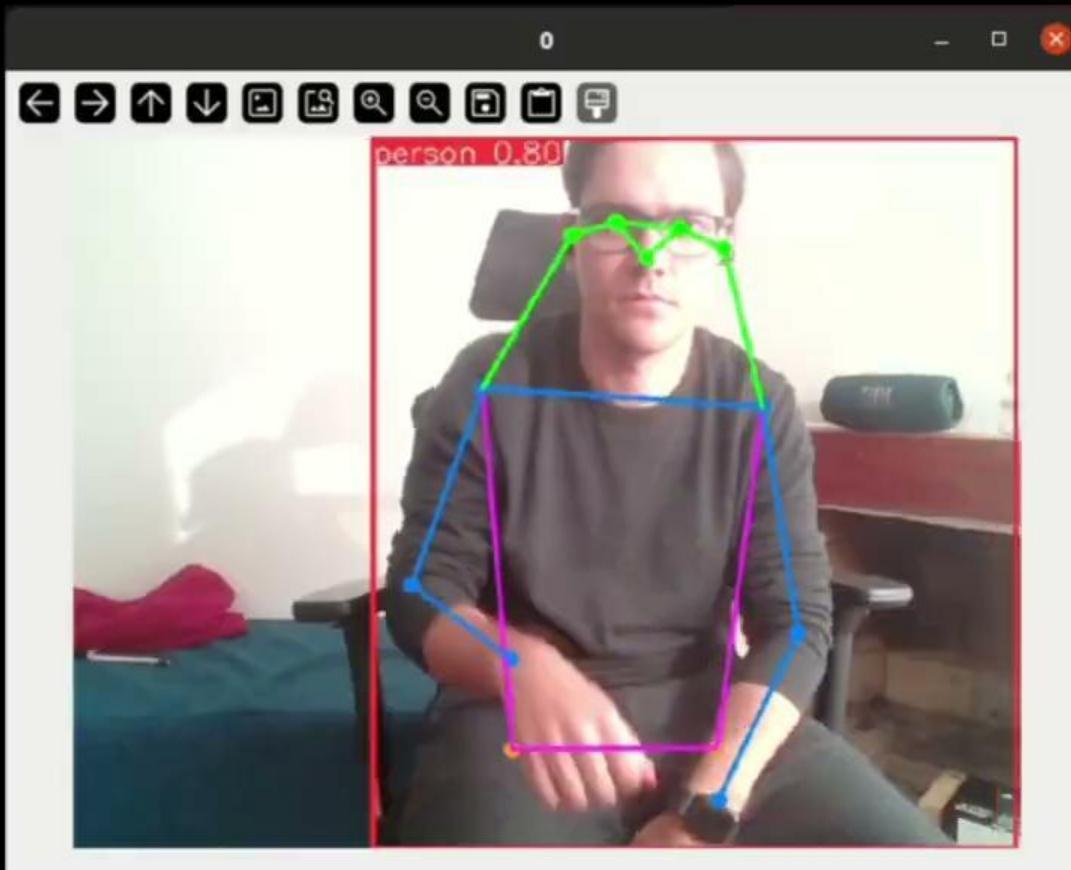
Snap

Container

Cameras

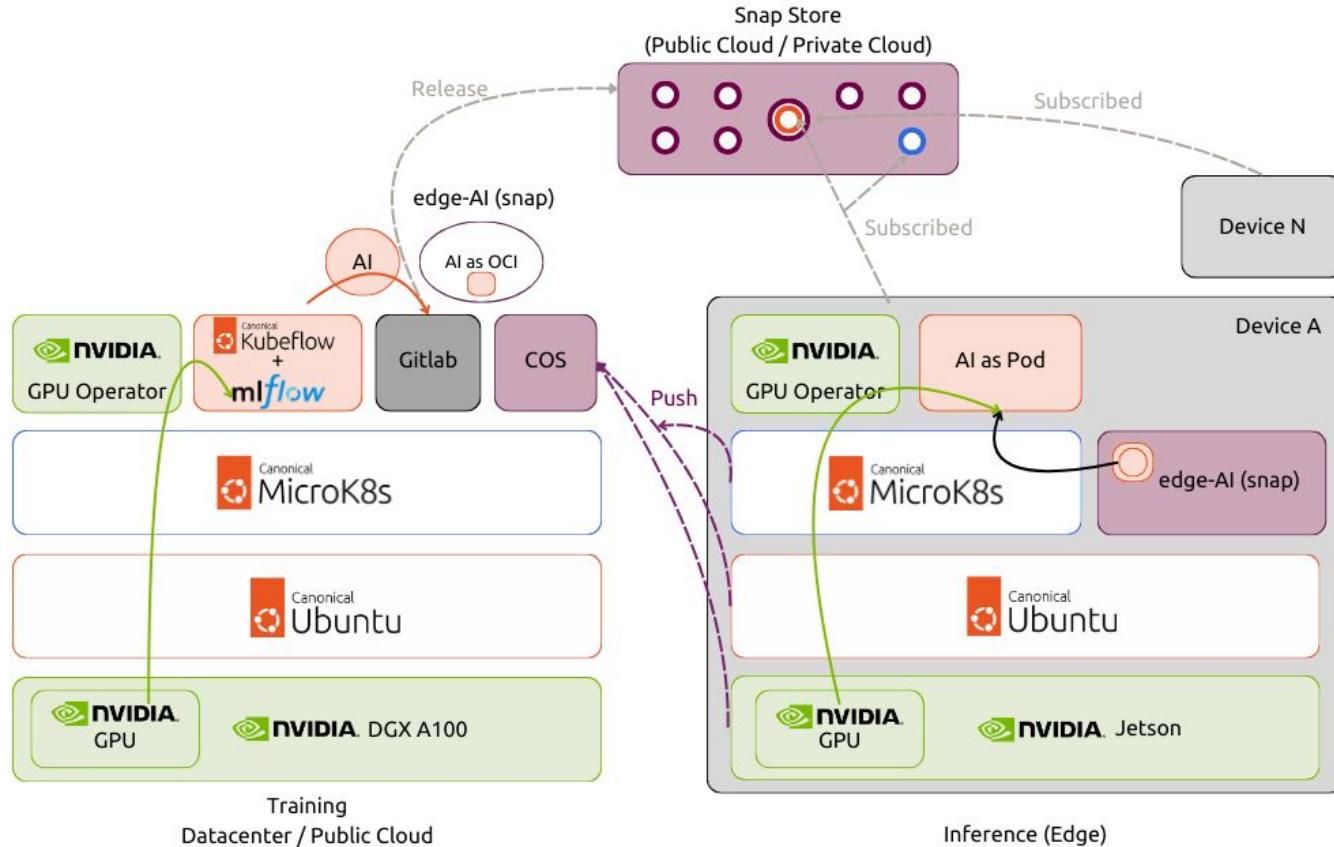
Inference

Edge



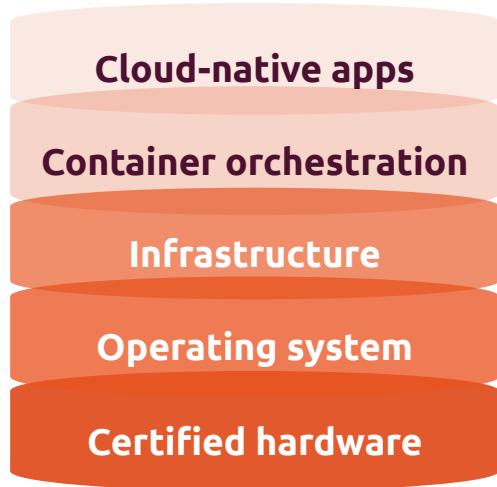


Open-source architecture





An end-to-end solution For scalable smart cities



NVIDIA



Canonical

Meet with us at booth
1 6 0 1

Let's connect

LinkedIn



Joint webinar on April 4th

Ubuntu on Jetson: A Journey From Development to Production



Canonical

20 Years of innovation