UNIVERSITY OF WESTMINSTER

Building a Cloud Toolkit

Jay DesLauriers

UNIVERSITY OF
LEADING
THE WAY
WESTMINSTER

The Research Centre for Parallel Computing @ UoW

Projects in Distributed Computing, from Grids to Cloud (to Fog/Edge)

Well-funded by EU/UK Research Grants (>£2.5mil since 2015)

- EDGeS: Enabling Desktop Grids for e-Science (2008)
- EDGI: European Desktop Grid Initiative (2010)
- VENUS-C: Virtual multidisciplinary environments using Cloud Infrastructures (2012)
- CloudSME: Cloud based Simulation Platform for Manufacturing & Engineering (2013)
- **COLA: Cloud Orchestration at the Level of Application (2017)**
- ASCLEPIOS: Advanced Secure Cloud Encrypted Platform for Internationally Orchestrated Solutions in Healthcare (2018)

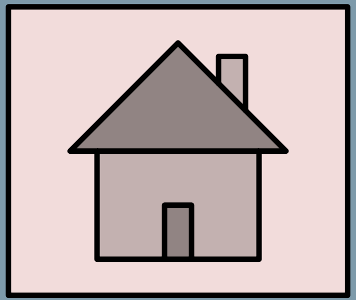
UNIVERSITY OF LEADING THE WAY WESTMINSTER

Why Cloud?

It's disruptive. Compute now available "as-a-Service"

No upfront cost for hardware or software licenses

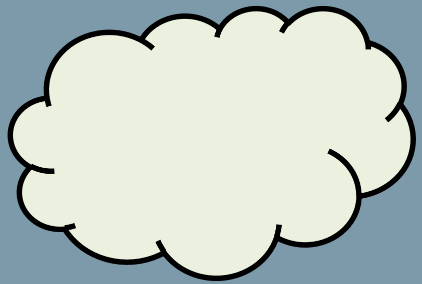
No operating or maintenance cost for local IT infrastructure



On-Premise

Capital expense model

££££



Cloud

Pay-as-you-go model

££

Reality-Check

Take-up still relatively low for research applications & by small business

Vendor lock-in: going multi-cloud is expensive, complex or both

Application-level auto-scaling is limited

Issues of security, privacy and trust

Project COLA

EU Horizon2020 Programme for Research & Innovation

33 months, 14 partners, 6 countries

Secure, cloud agnostic application-level auto-scaling
to encourage cloud uptake

UNIVERSITY OF LEADING THE WAY WESTMINSTER

MiCADO

MiCADO

Microservice-based Cloud Application-level Dynamic Orchestrator

Terraform

Kubernetes

Prometheus

UNIVERSITY OF WESTMINSTER — Interface

BALASYS — Security

MTA SZTAKI — Scaling

Cloud Orchestration

Infrastructure-as-a-Service (IaaS)

- Provisioning virtual machines from a cloud service provider (CSP)

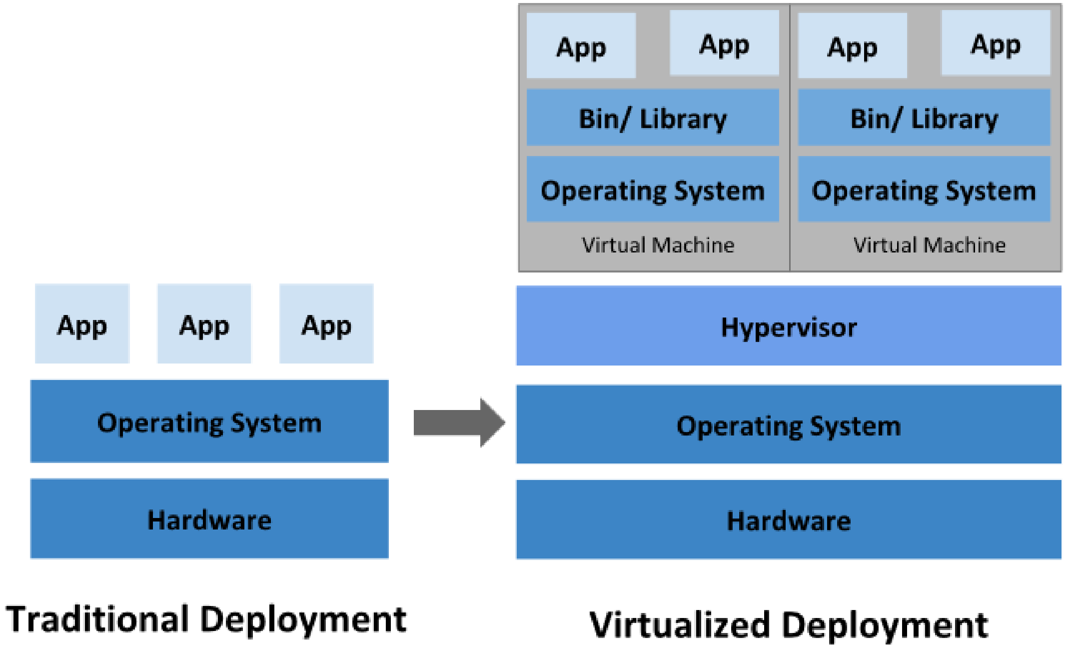
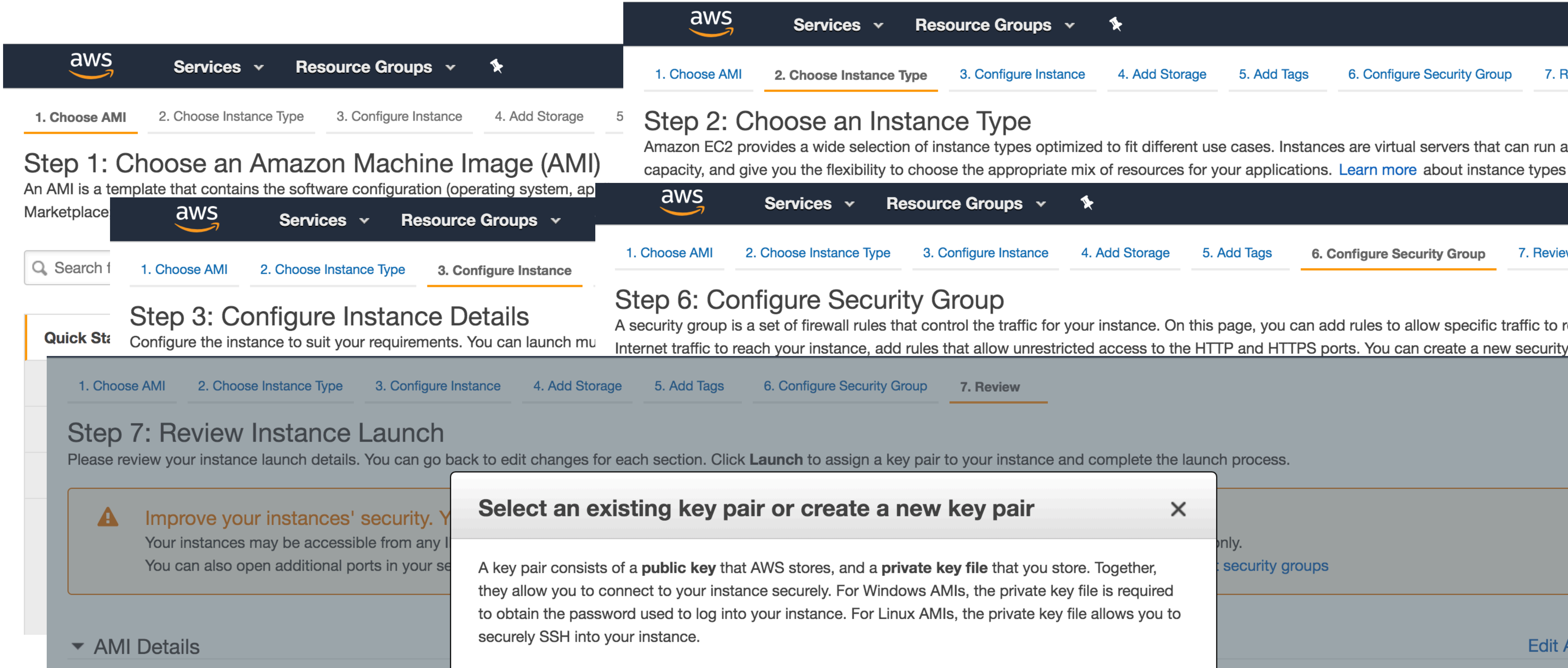


Image Source: https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/

Manual Cloud Provisioning



UNIVERSITY OF
LEADING
THE WAY
WESTMINSTER

Terraform

Infrastructure-as-Code

```
provider "aws" {
  region = "us-west-2"
}

data "aws_ami" "ubuntu" {
  most_recent = true

  filter {
    name   = "name"
    values = ["ubuntu/images/hvm-ssd/ubuntu-t
  }

}

resource "aws_instance" "web" {
  ami           = "${data.aws_ami.ubuntu.id}"
  instance_type = "t2.micro"

  tags = {
```

```
TERMINAL > terraform plan
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will
persisted to local or remote state storage.

------------------------------------------------------------------------

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_instance.web will be created
  + resource "aws_instance" "web" {
      + ami                          = "ami-0bac6fc47ad07c5f5"
```

UNIVERSITY OF
LEADING
THE WAY
WESTMINSTER

Terraform

Infrastructure Management

–Provisions
–Maintains
–Destroys

–Scales
–Self-healing

UNIVERSITY OF WESTMINSTER
LEADING THE WAY

Container Orchestration

Application containers

- Lightweight OS-virtualisation
- Application packaging for portable, reusable software

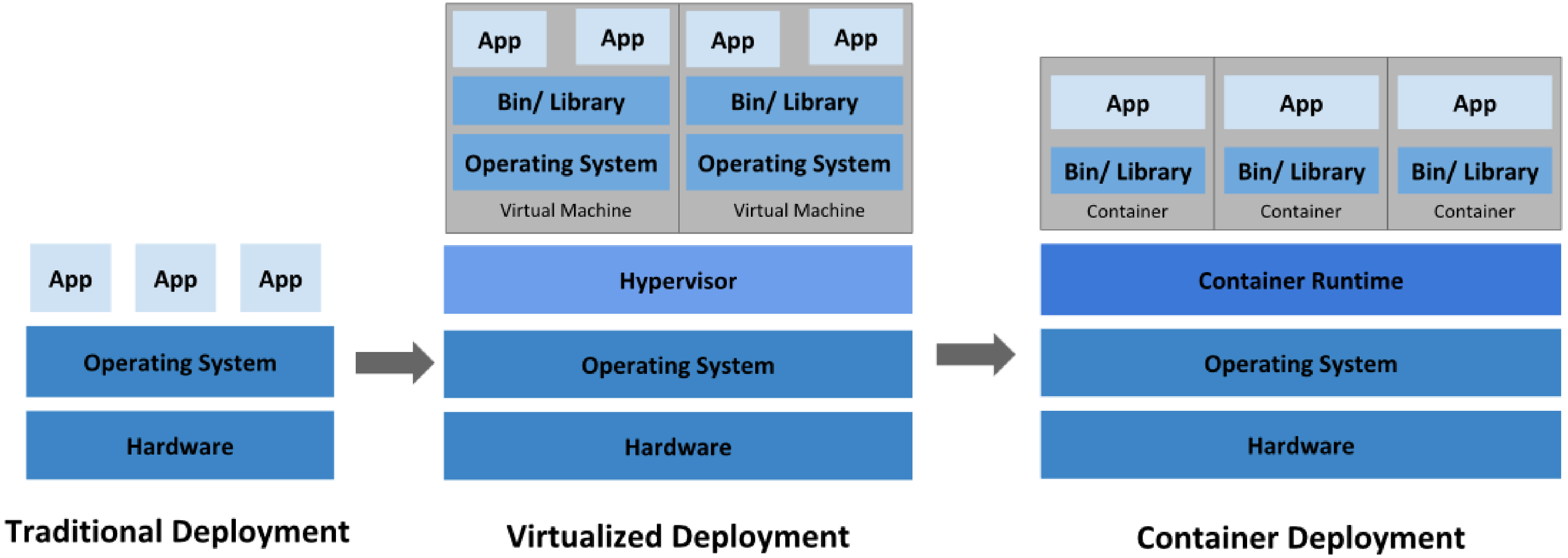


Image Source: https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/

Container Orchestration

```
TERMINAL > docker run busybox cal -j
       December 2019
 Su  Mo  Tu  We  Th  Fr  Sa
335 336 337 338 339 340 341
342 343 344 345 346 347 348
349 350 351 352 353 354 355
356 357 358 359 360 361 362
363 364 365

TERMINAL > docker run -d busybox sleep 60
211685b29840d758974795a662b14c1d6df807ec792faed90fc84b0557b84e5b

TERMINAL > docker ps
CONTAINER ID        IMAGE                                       COMMAND
68c958637f70        busybox                                     "sleep 60"
```

UNIVERSITY OF
LEADING
THE WAY
WESTMINSTER

Kubernetes

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: myapp-pod
  labels:
    app: myapp
spec:
  containers:
  - name: myapp-container
    image: busybox
    command: ['sh', '-c', 'echo Hello Kubernetes
```

```
TERMINAL > vim kube-test.yaml
kubectl apply -f kube-test.yaml
pod/myapp-pod created
TERMINAL > kubectl get po
NAME        READY   STATUS    RESTARTS   AGE
myapp-pod   1/1     Running   0          6s
TERMINAL >
```

Kubernetes

Container Management Across a Cluster of Nodes (VMs)

– Deploys
– Maintains
– Destroys

– Auto-scaling
– Self-healing
– Rolling updates and rollbacks CI/CD

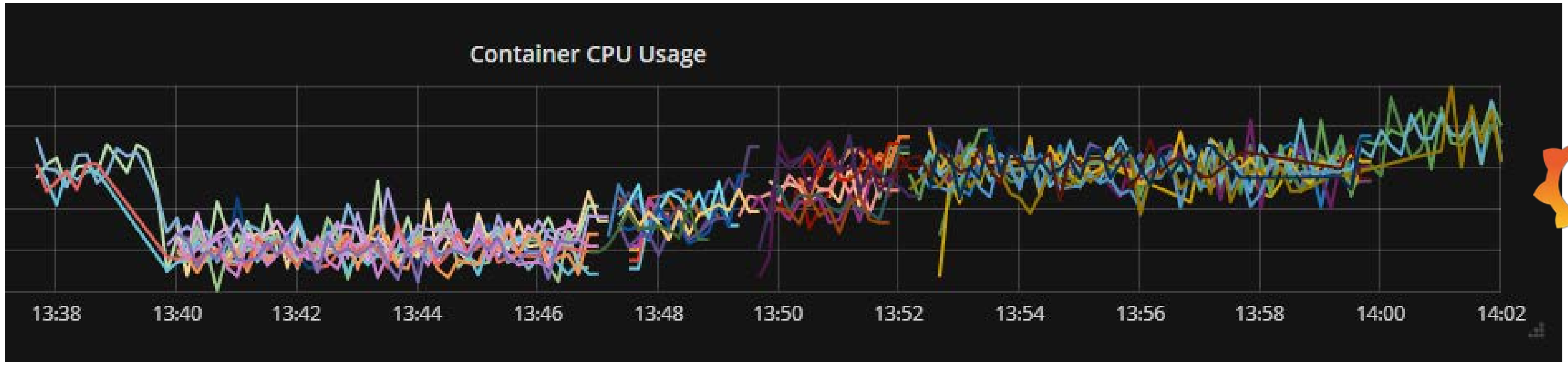
Prometheus

A monitoring & alerting system

Pull-style metric collection

- Resource usage of containers / virtual machines (CPU, Memory, etc…)
- Custom data exported by applications (latency, requests served)

Alerting based on those metrics

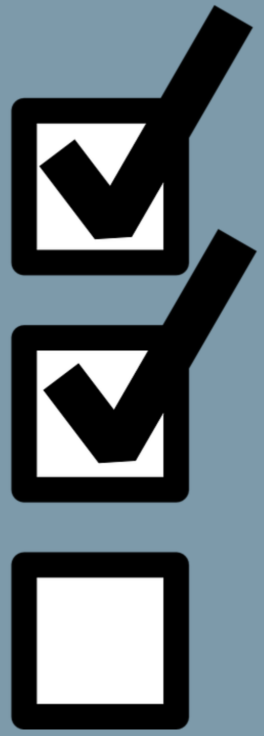


UNIVERSITY OF
LEADING
THE WAY
WESTMINSTER

Terraform, Kubernetes & Prometheus for Research

Some good things:
- Open-Source
- Community
- Extensions
  - Kubernetes (Kubeflow)
  - Prometheus Exporters (DBs)
  - Terraform Modules (Sagewatch, BigQuery)

Could-be-better things:
- High overall complexity
  - Deploying, configuring, integrating
- Vendor lock-in encouraged
- Limited scope for auto-scaling



UNIVERSITY OF
LEADING
THE WAY
WESTMINSTER

MiCADO

Microservice-based Cloud Application-level Dynamic Orchestrator

ASSEMBLY REQUIRED

Cloud Orchestration

Container Orchestration

Monitoring

UNIVERSITY OF WESTMINSTER

BALASYS

MTA SZTAKI

Interface

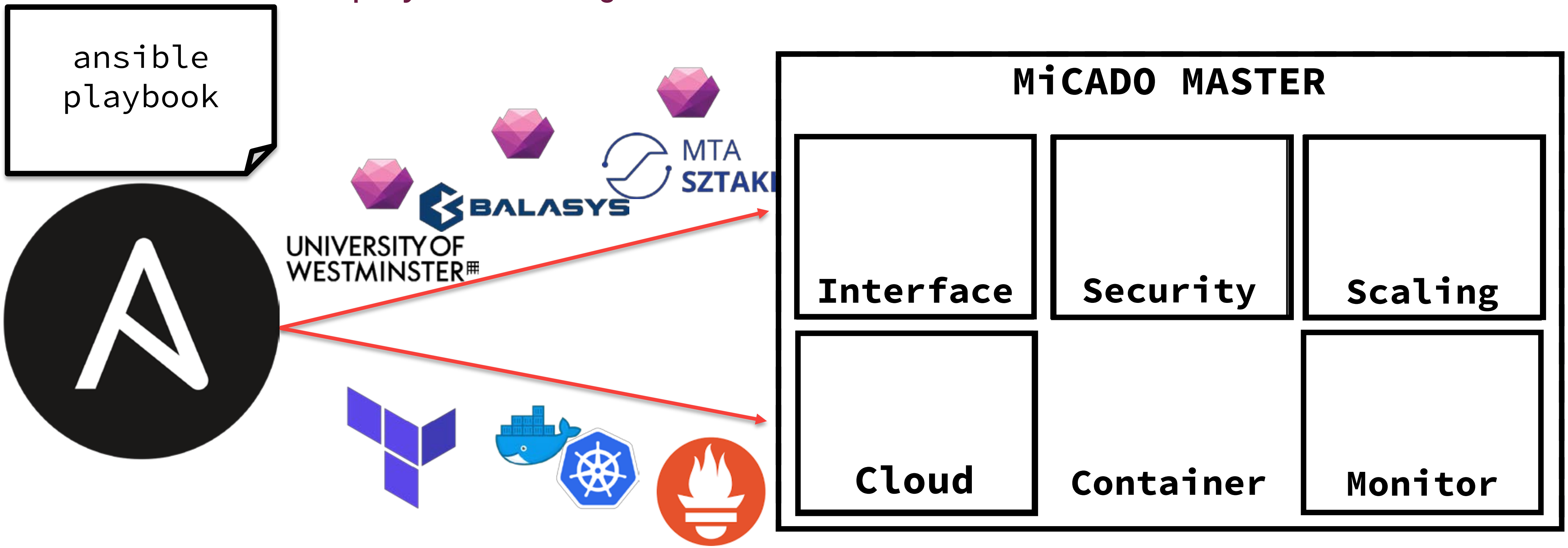
Security

Scaling

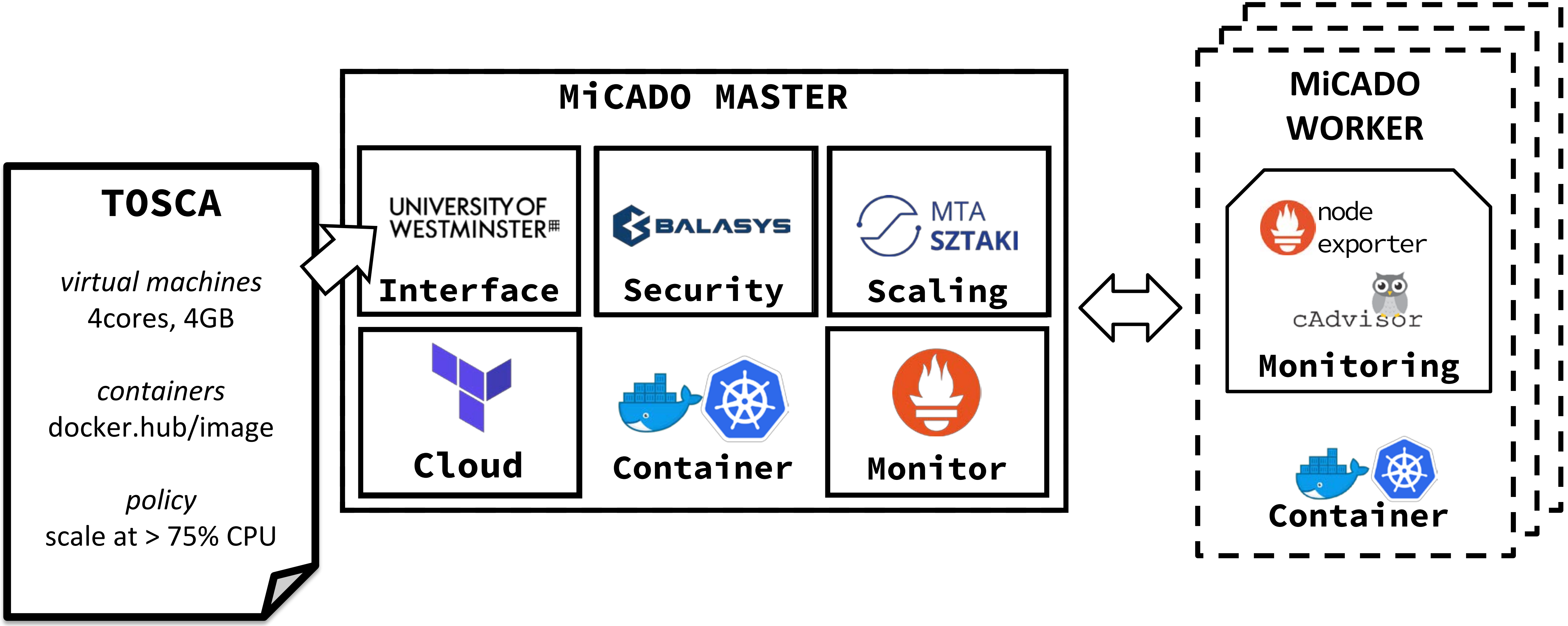
Ansible

Declarative configuration management automation framework

Deploys and configures MiCADO microservices



MiCADO



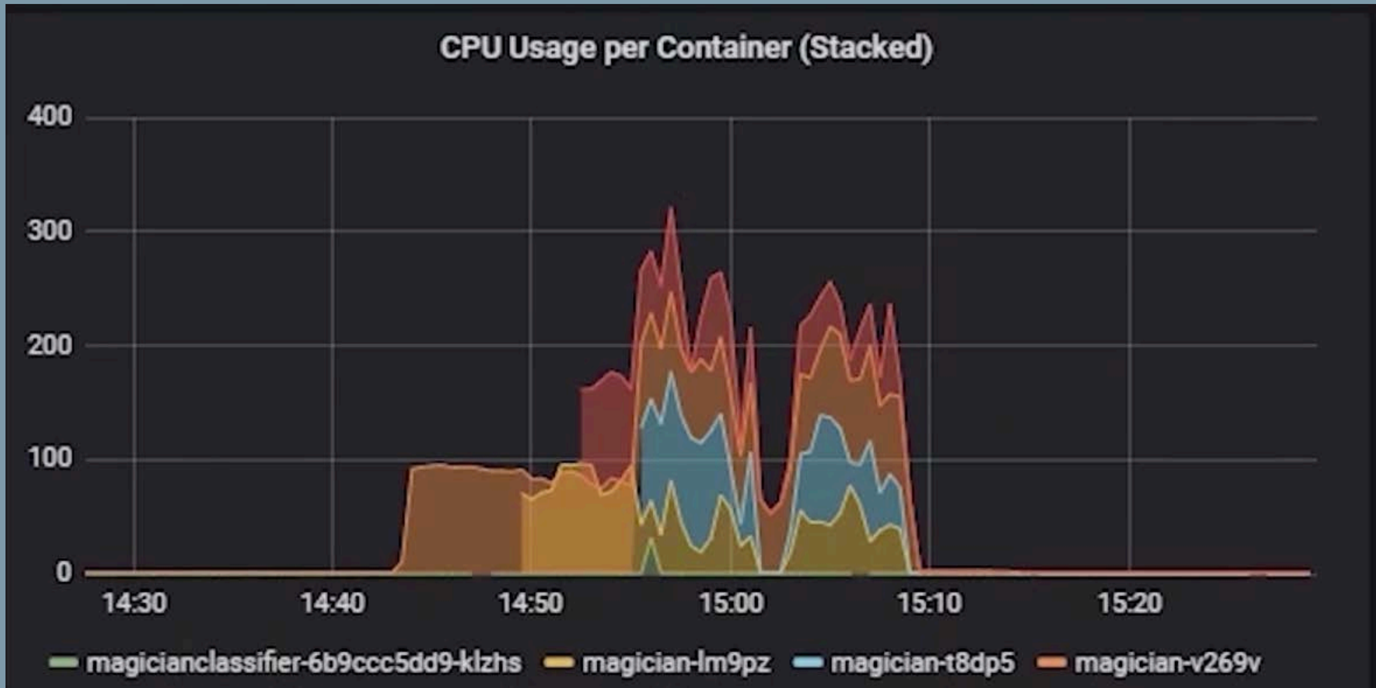
TOSCA
virtual machines
4cores, 4GB
containers
docker.hub/image
policy
scale at > 75% CPU
MiCADO MASTER
UNIVERSITY OF WESTMINSTER
Interface
BALASYS
Security
MTA SZTAKI
Scaling
Cloud
Container
Monitor
MiCADO WORKER
node exporter
cAdvisor
Monitoring
Container

Social Media Analytics Use-Case

- Resource intensive services
  - Typically CPU/memory –bound apps/services
  - Containers & underlying VMs scale to meet demand

GOBIERNO DE ARAGON

Inycom

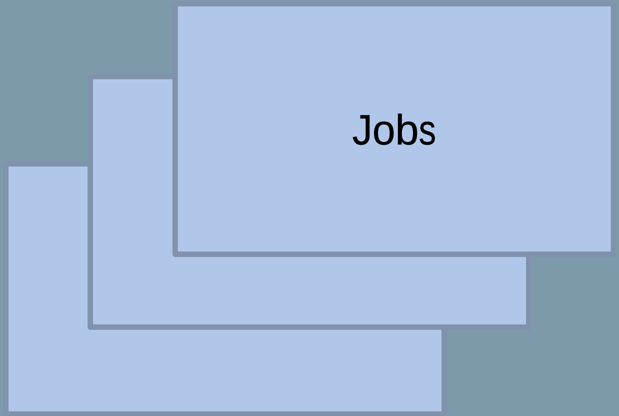


UNIVERSITY OF
LEADING
THE WAY
WESTMINSTER

Simulation & Modelling Use-Case

- Multi-job, deadline constrained experiments
  - Typically batch/parameter sweep jobs
  - Containers/VMs scale to complete jobs by deadline

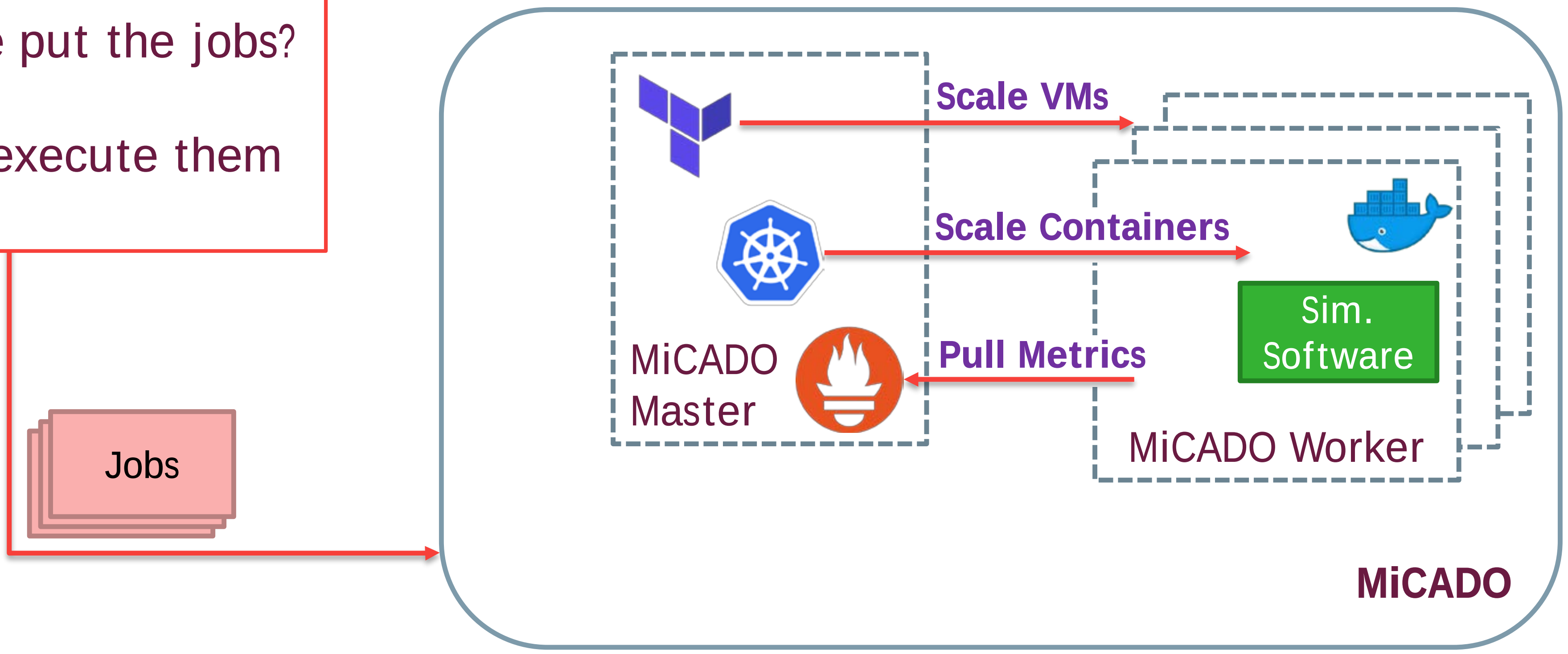
Brunel University London



Insert Queue Here

Where do we put the jobs?

How do we execute them



jQueuer

Asynchronous Distributed Task Queue

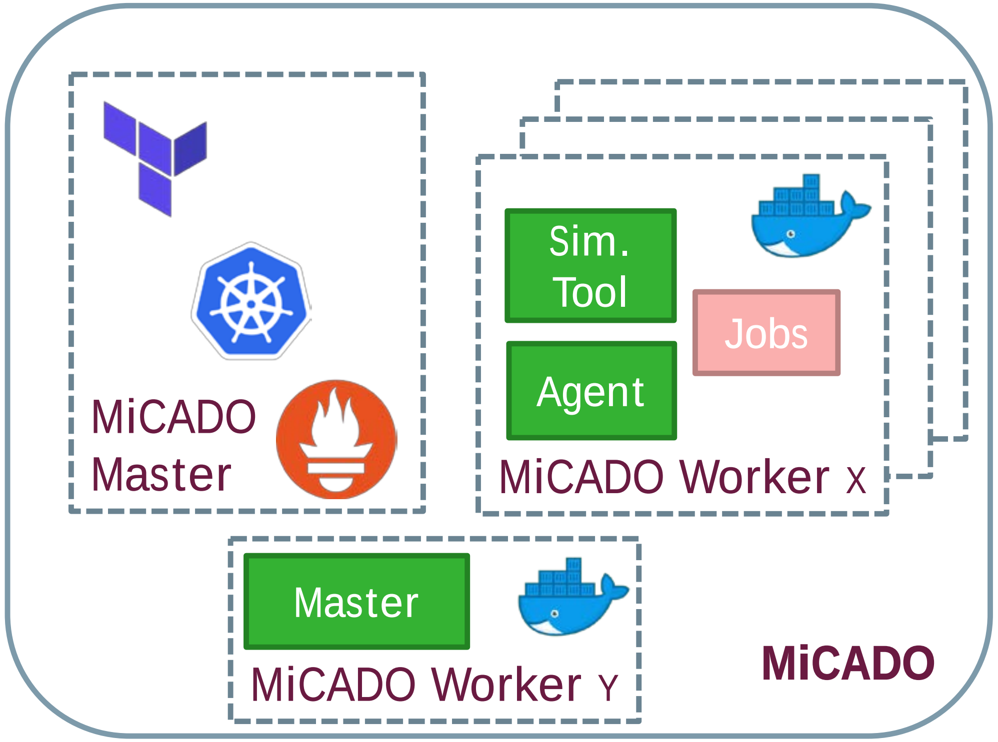
- Celery.py

Master Component

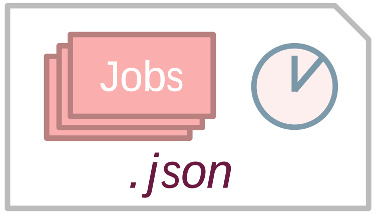
- The queue
- Metric Generation
- Frontend for submission

Agent Component

- Runs alongside experiment tool
- Fetches jobs from Master
- Executes jobs in container



- JSON input
  - Jobs & Deadline



Deadline-based auto-scaling

Calculates containers/VMs required to complete jobs by deadline

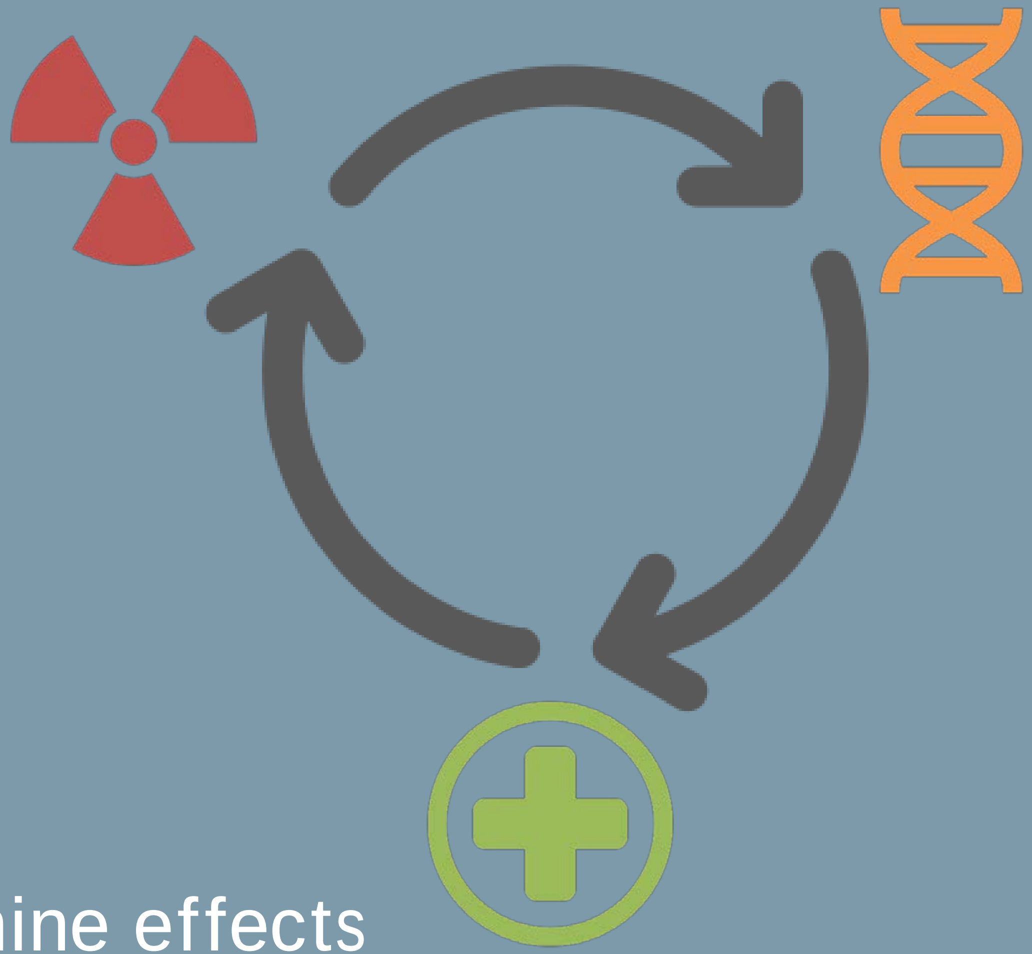
Uses jQueuer metrics:
- Queue length
- Jobs completed
- Jobs remaining
- Time elapsed
- Average job length
- Time to deadline

Cloud resources are scaled up/down by MiCADO

UNIVERSITY OF LEADING THE WAY WESTMINSTER

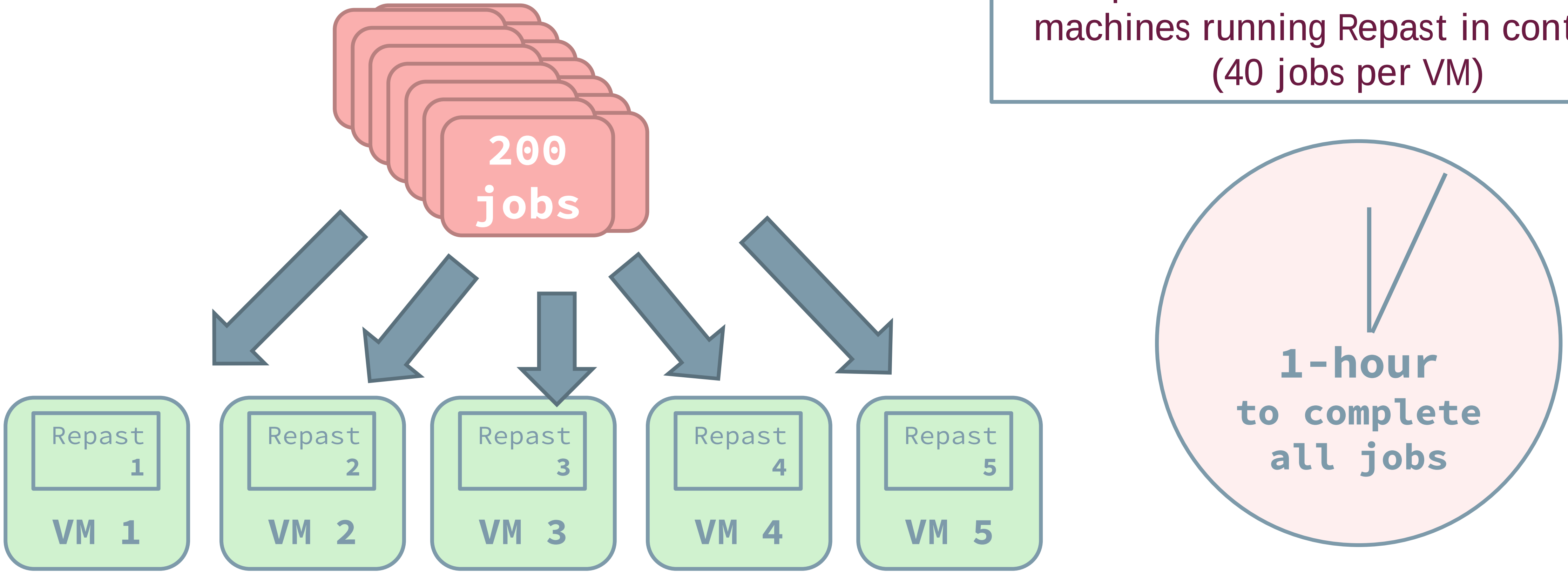
THE EXPERIMENT

- Agent-based simulation
  - Repast Simphony
- Three agents
  - Infected
  - Susceptible
  - Recovered
- Simulate movement & interaction of agents in an environment to determine effects of one group on another

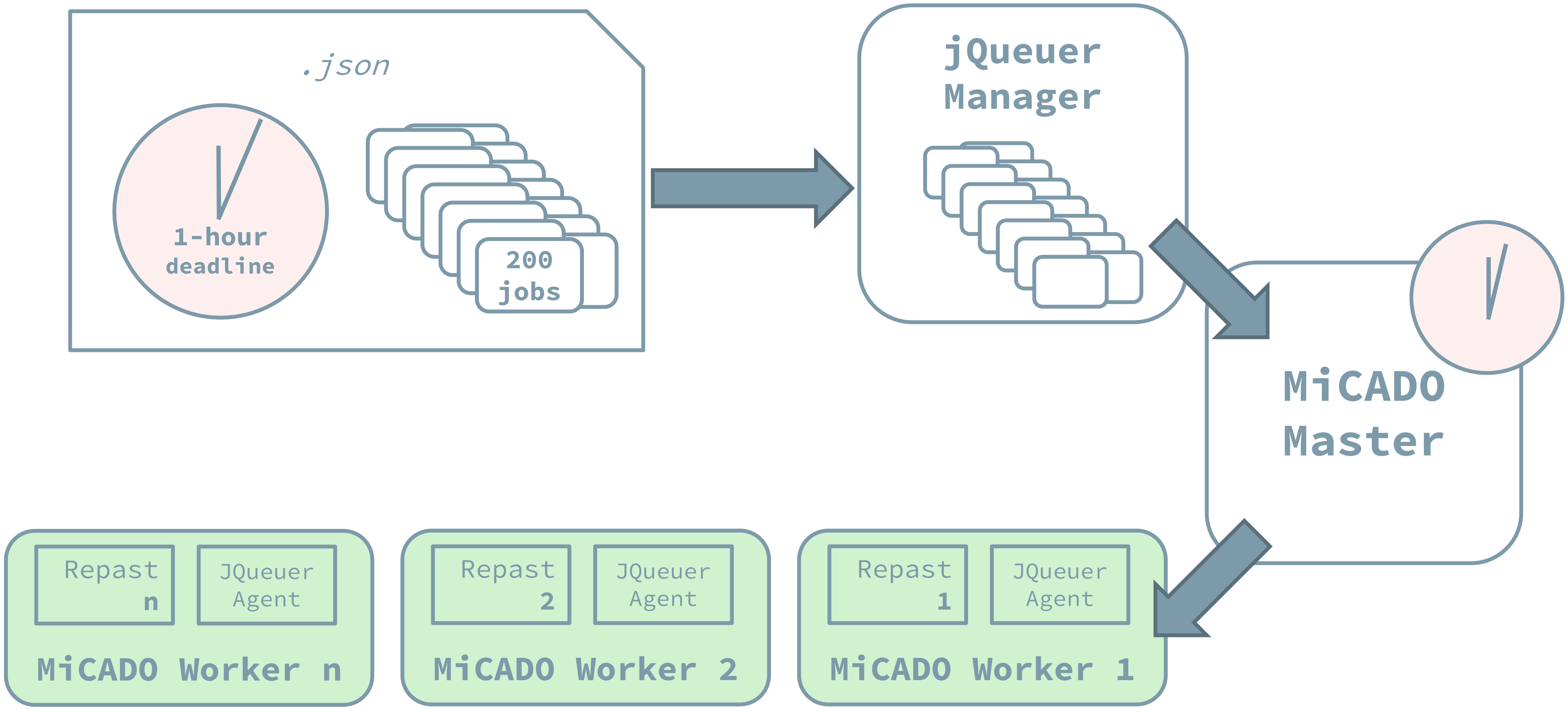


Calculating a Baseline: Manual allocation

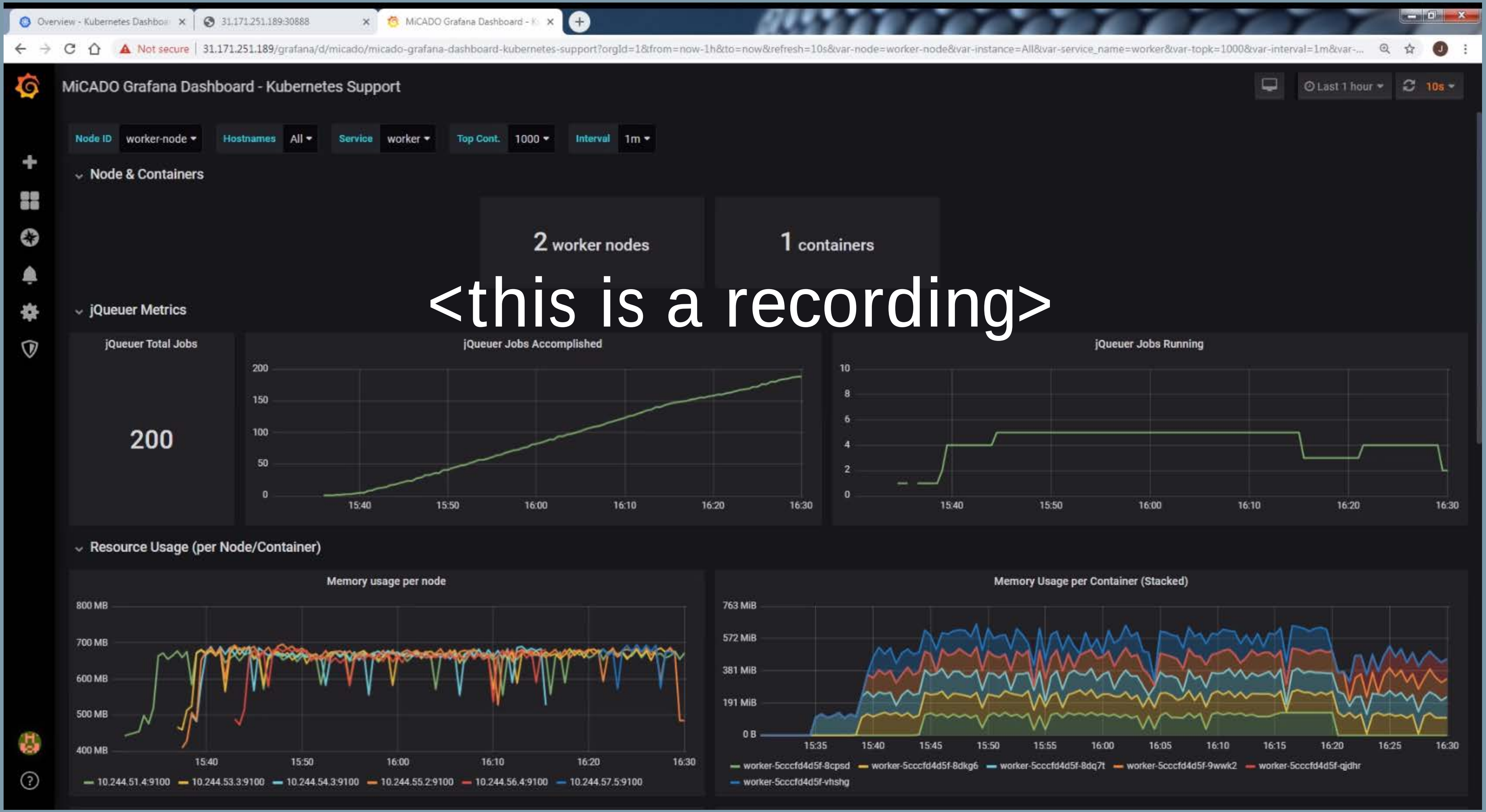
Equal distribution to five virtual machines running Repast in container (40 jobs per VM)



Using MiCADO: Dynamic allocation & auto-scaling



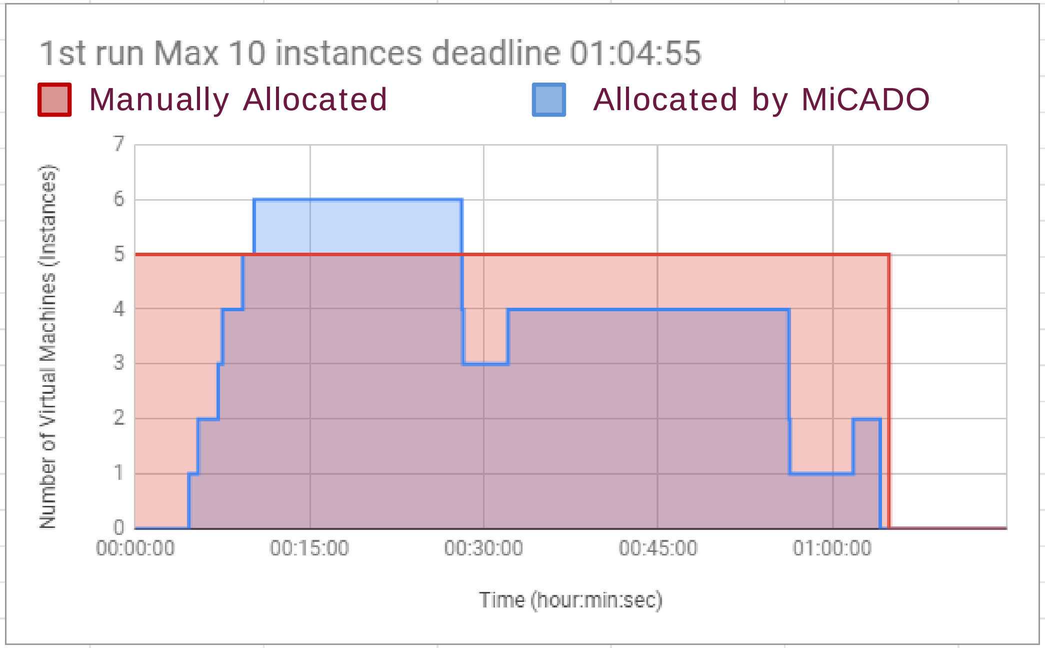
UNIVERSITY OF
LEADING
THE WAY
WESTMINSTER

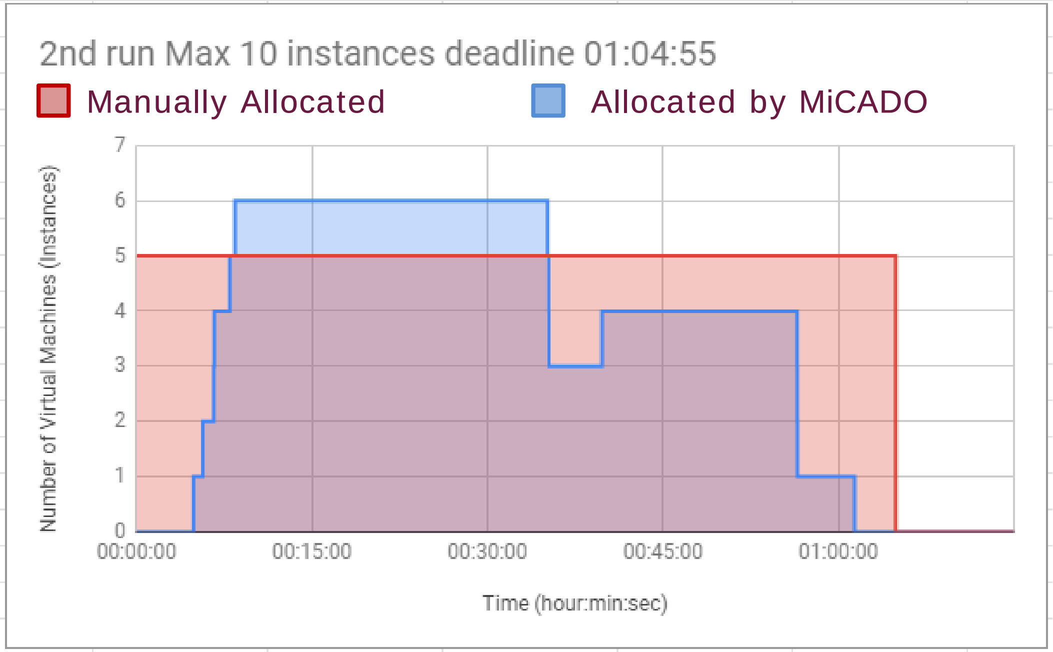


<this is a recording>

Results

Dynamic allocation of variable length jobs results in a better use of cloud resources





5 VMs

Manual allocation (baseline)

3.86 VMs

Dynamic allocation (MiCADO)

UNIVERSITY OF LEADING THE WAY WESTMINSTER

Cast (in order of appearance)

Terraform terraform.io

Kubernetes kubernetes.io

Prometheus prometheus.io

Ansible ansible.io

MiCADO micado-scale.eu

jQueuer doi.org/10.1016/j.future.2019.05.062

Thanks!

Jay DesLauriers
j.deslauriers@westminster.ac.uk