



TOSCA Orchestration

Advanced usage



eosc-hub.eu



[@EOSC_eu](https://twitter.com/EOSC_eu)

Dissemination level: Public

Marica Antonacci

INFN BARI

marica.antonacci@ba.infn.it

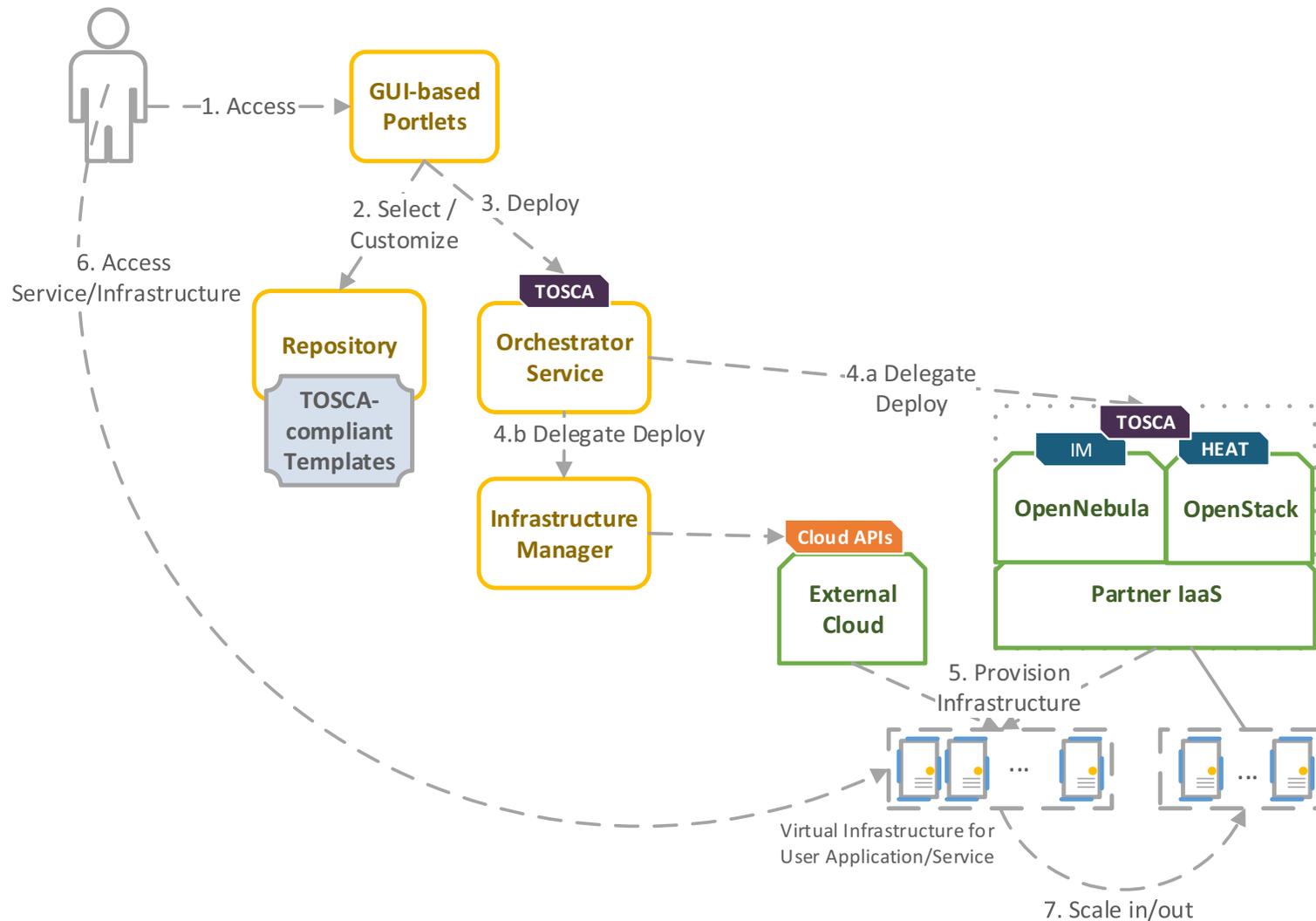
Istanziamento e utilizzo di batch system on demand su infrastrutture Cloud

25-28 November 2019 - Perugia (Italy)

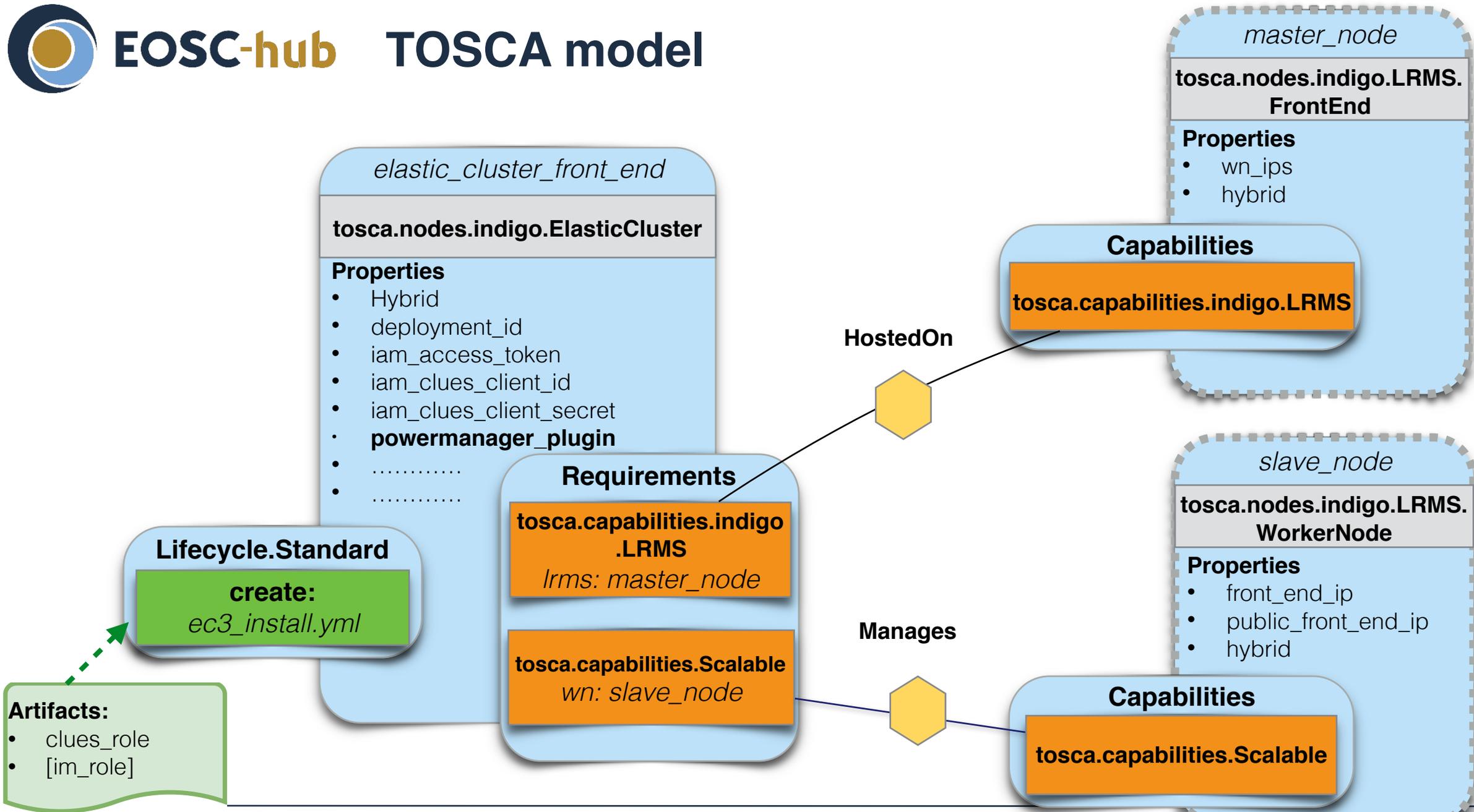


- TOSCA Orchestration Advanced Use-cases
 - Elastic Clusters
 - Laniakea: Galaxy on demand
 - Big Data Analysis Facility
 - Managed Long Running Services
 - DEEPaaS for ML
 - HPC job submission from PaaS

EOSC-hub Deployment of Elastic Clusters



- CLUES is a tool that **powers off** those cluster **nodes** that are **not** being **used**, and powers them on when demanded and in real time.
- CLUES integrates with the existing local resource manager and carries out its activities transparently to the final user.
- When a user requests the execution of a job (by sending it to a batch queue), CLUES checks if the currently available nodes will be able to process the request. If the number of resources is not enough, CLUES will try to power on the nodes that are necessary for the execution of the task.



indigo.LRMS.FrontEnd

indigo.LRMS.FrontEnd.Kubernetes

indigo.LRMS.FrontEnd.Mesos

indigo.LRMS.FrontEnd.Slurm

indigo.LRMS.FrontEnd.Torque

indigo.LRMS.WorkerNode

indigo.LRMS.WorkerNode.Kubernetes

indigo.LRMS.WorkerNode.Mesos

indigo.LRMS.WorkerNode.Slurm

indigo.LRMS.WorkerNode.Torque

- The base type defines the "create" interface

```
---  
- hosts: localhost  
  connection: local  
  roles:  
  - role: grycap.im  
    when: clues_powermanager_plugin == "im"  
  - role: indigo-dc.clues  
    indigo_orchestrator_url: "{{ orchestrator_url }}"
```



When using the orchestrator the clues_powermanager_plugin is set to indigo_orchestrator

```
elastic_cluster_front_end:
  type: toska.nodes.indigo.ElasticCluster
  properties:
    deployment_id: orchestrator_deployment_id
    iam_access_token: iam_access_token
    iam_clues_client_id: iam_clues_client_id
    iam_clues_client_secret: iam_clues_client_secret
    hybrid: { get_input: hybrid }
  requirements:
    - lrms: lrms_front_end
    - wn: wn_node
    - host: lrms_server

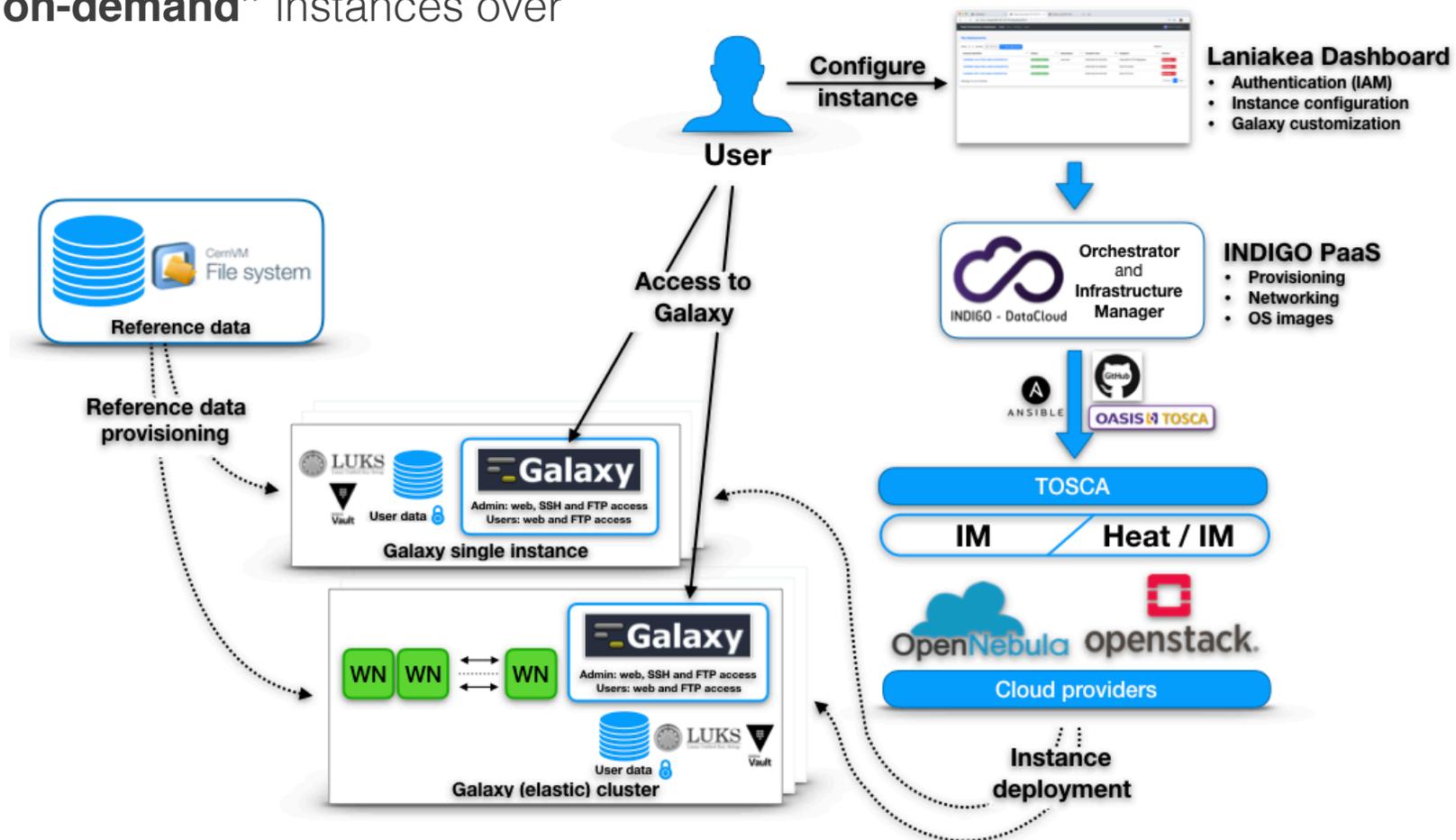
lrms_front_end:
  type: toska.nodes.indigo.LRMS.FrontEnd.Slurm
  properties:
    wn_ips: { get_attribute: [ lrms_wn, private_address ] }
    hybrid: { get_input: hybrid }
  requirements:
    - host: lrms_server
```

```
wn_node:
  type: toska.nodes.indigo.LRMS.WorkerNode.Slurm
  properties:
    front_end_ip: { get_attribute: [ lrms_server, private_address, 0 ] }
    public_front_end_ip: { get_attribute: [ lrms_server, public_address, 0 ] }
    hybrid: { get_input: hybrid }
  capabilities:
    wn:
      properties:
        min_instances: 1
        max_instances: { get_input: wn_num }
        default_instances: 1
  requirements:
    - host: lrms_wn
```

An open solution to provide **Galaxy** “on-demand” instances over heterogeneous cloud infrastructures.

Developed by CNR-IBIOM and INFN

It's built on the INDIGO PaaS



doi: <https://doi.org/10.1101/472464>

Credits: M. Tangaro (CNR-IBIOM)

<https://www.youtube.com/watch?v=cq-68-2On3Y&feature=youtu.be>

```
elastic_cluster_front_end:
  type: tosca.nodes.indigo.ElasticCluster
  properties:
    deployment_id: orchestrator_deployment_id
    iam_access_token: iam_access_token
    iam_clues_client_id: iam_clues_client_id
    iam_clues_client_secret: iam_clues_client_secret
  requirements:
    - lrms: lrms_front_end
    - wn: wn_node

galaxy_portal:
  type: tosca.nodes.indigo.GalaxyPortal
  properties:
    admin_email: { get_input: admin_email }
    admin_api_key: { get_input: admin_api_key }
    version: { get_input: version }
    instance_description: { get_input: instance_description }
    instance_key_pub: { get_input: instance_key_pub }
  requirements:
    - lrms: lrms_front_end
```

```
lrms_front_end:
  type: tosca.nodes.indigo.LRMS.FrontEnd.Slurm
  properties:
    wn_ips: { get_attribute: [ lrms_wn, private_address ] }
  requirements:
    - host: lrms_server
```

```
wn_node:
  type: tosca.nodes.indigo.LRMS.WorkerNode.Slurm
  properties:
    front_end_ip: { get_attribute: [ lrms_server, private_address, 0 ] }
  capabilities:
    wn:
      properties:
        max_instances: { get_input: wn_num }
        min_instances: 0
  requirements:
    - host: lrms_wn

galaxy_wn:
  type: tosca.nodes.indigo.GalaxyWN
  requirements:
    - host: lrms_wn
```

elastic_cluster_front_end:

```

type: toska.nodes.indigo.ElasticCluster
properties:
  deployment_id: orchestrator_deployment_id
  iam_access_token: iam_access_token
  iam_clues_client_id: iam_clues_client_id
  iam_clues_client_secret: iam_clues_client_secret
  marathon_credentials:
    protocol: https
    token: { get_input: marathon_password }
    user: admin
  chronos_credentials:
    protocol: https
    token: { get_input: chronos_password }
    user: admin
  mesos_credentials:
    protocol: http
    token: { get_input: mesos_password }
    user: admin
  requirements:
    - lrms: mesos_master
    - wn: mesos_slave

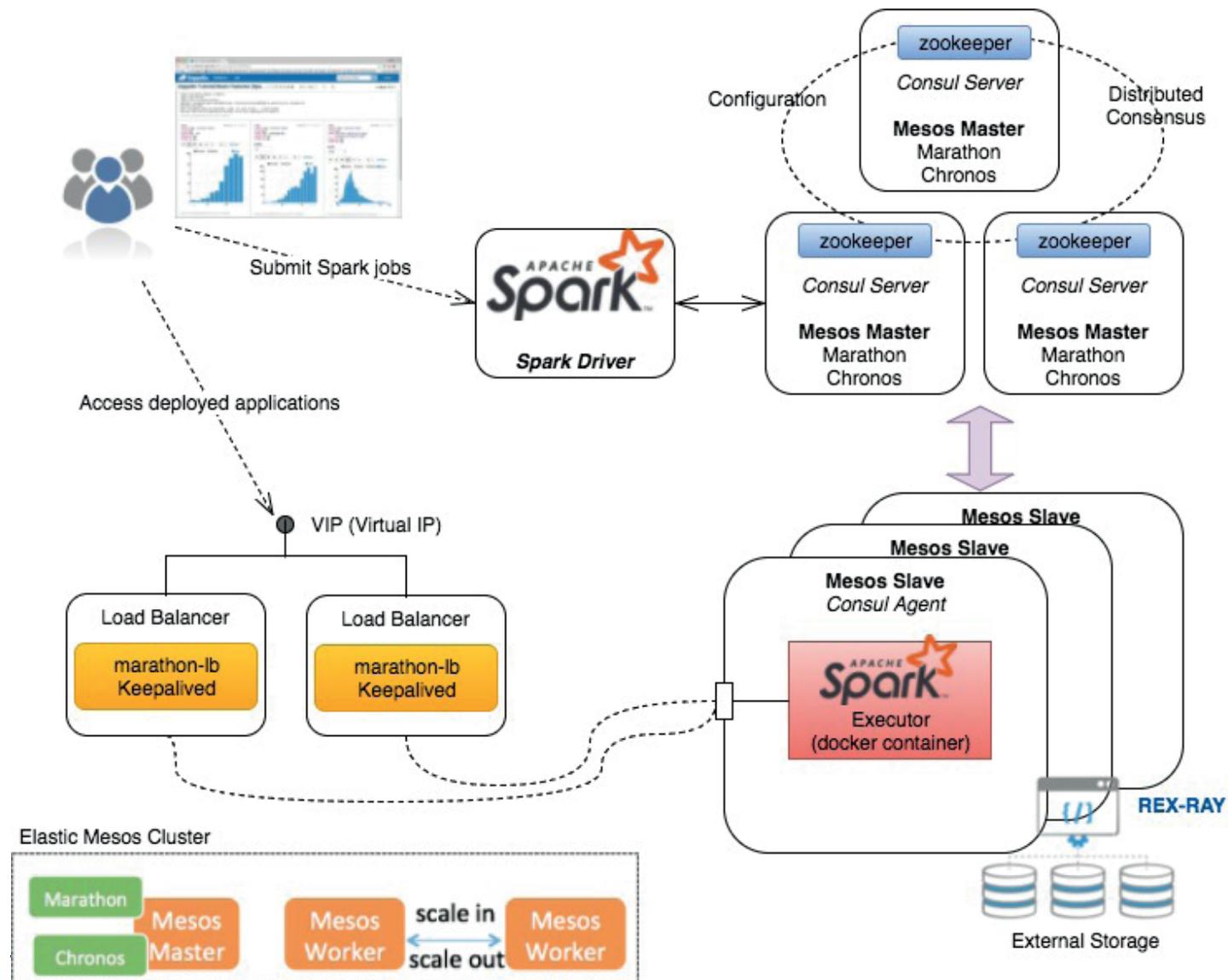
```

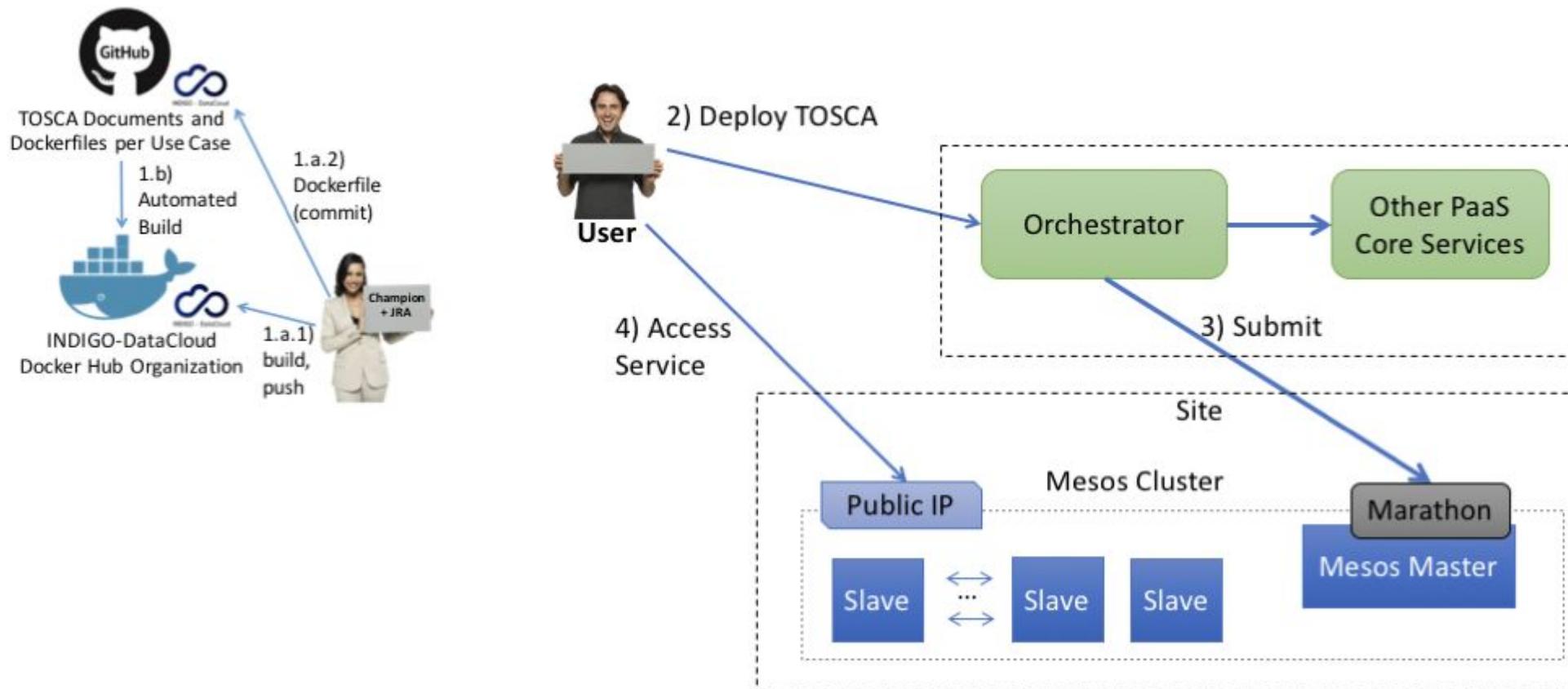
mesos_master:

```

type: toska.nodes.indigo.LRMS.FrontEnd.Mesos
properties:
  mesos_masters_list: { get_attribute: [ HOST, private_address ] }
  mesos_password: { get_input: mesos_password }
  marathon_password: { get_input: marathon_password }
  chronos_password: { get_input: chronos_password }
requirements:
  - host: mesos_master_server

```



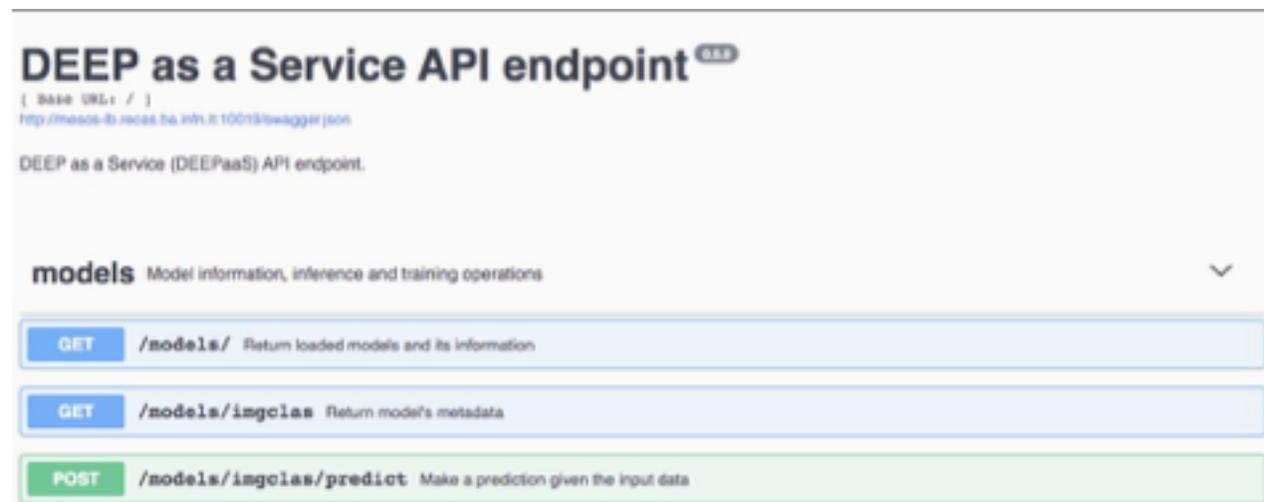


```
node_templates:

marathon-app:
  type: tosca.nodes.indigo.Container.Application.Docker.Marathon
  properties:
    # environment_variables:
  artifacts:
    image:
      file: { get_input: docker_image }
      type: tosca.artifacts.Deployment.Image.Container.Docker
  requirements:
    - host: docker_runtime

docker_runtime:
  type: tosca.nodes.indigo.Container.Runtime.Docker
  capabilities:
    host:
      properties:
        num_cpus: { get_input: cpus }
        mem_size: { get_input: mem }
        publish_ports:
          - protocol: tcp
            source: { get_input: port }
        volumes: [ { concat: [ 'marathon:', get_input: data_path, ':rw:dvdi:rexray' ] } ]
```

- Deep Learning prediction modules included in the DEEP-HybridDataCloud Open Catalog (<https://marketplace.deep-hybrid-datacloud.eu/>) can be deployed through a TOSCA template
 - the DEEPaaS API is deployed as long running service on Mesos cluster
 - the API can be accessed from the web browser and used to make predictions



https://youtu.be/Ax0Kaw01X7Q?list=PLJ9x9Zk1O-J_UZfNO2uWp2pFMmbwLvzXa

```

tosca.nodes.indigo.Qcg.Job:
  derived_from: toska.nodes.indigo.Batch.Job
  properties:
    directory:
      required: no
      type: string
      description: Work directory for the Job
  schema:
    required: no
    type: string
    description: Job schema
  note:
    required: no
    type: string
    description: User's note for the Job
  
```

```

tosca.nodes.indigo.Batch.Job:
  derived_from: toska.nodes.Root
  properties:
    executable:
      required: yes
      type: string
      description: Name of the executable file
    arguments:
      required: no
      type: string
      description: Arguments for the job executable
    environment:
      type: map
      entry_schema:
        type: string
      description: A map of string representing environment settings
  
```

Work in progress



**Thank you
for your attention!**

Questions?



EOOSC-hub

 eosc-hub.eu  [@EOOSC_eu](https://twitter.com/EOOSC_eu)



This material by Parties of the EOOSC-hub Consortium is licensed under a [Creative Commons Attribution 4.0 International License](https://creativecommons.org/licenses/by/4.0/).