TOSCA-based orchestration of complex clusters at the IaaS level

Ricardo Rocha, German Molto, Mathieu Velten, Miguel Caballer, Giacinto Donvito
(on behalf of the Indigo DataCloud team)

INDIGO - DataCloud

RIA-653549

INDIGO-DataCloud is co-founded by the Horizon 2020Framework Programme

INDIGO DataCloud

- Towards a sustainable European SW infrastructure for e-science
- Funder under the European Union Horizon 2020 program
- Multiple partners, **heterogeneous infrastructures**
  - INFN, CERN, ATOS, UPV, CSIC, DESY, LIP, T-Systems, Reply, …
- Main strengths
  - Support for multi-disciplinary scientific communities
  - Exploits available, general solutions
  - Low learning curve, rely on popular software tools
  - Hybrid, distributed environment
- **IaaS**, PaaS, SaaS

https://www.indigo-datacloud.eu

‘

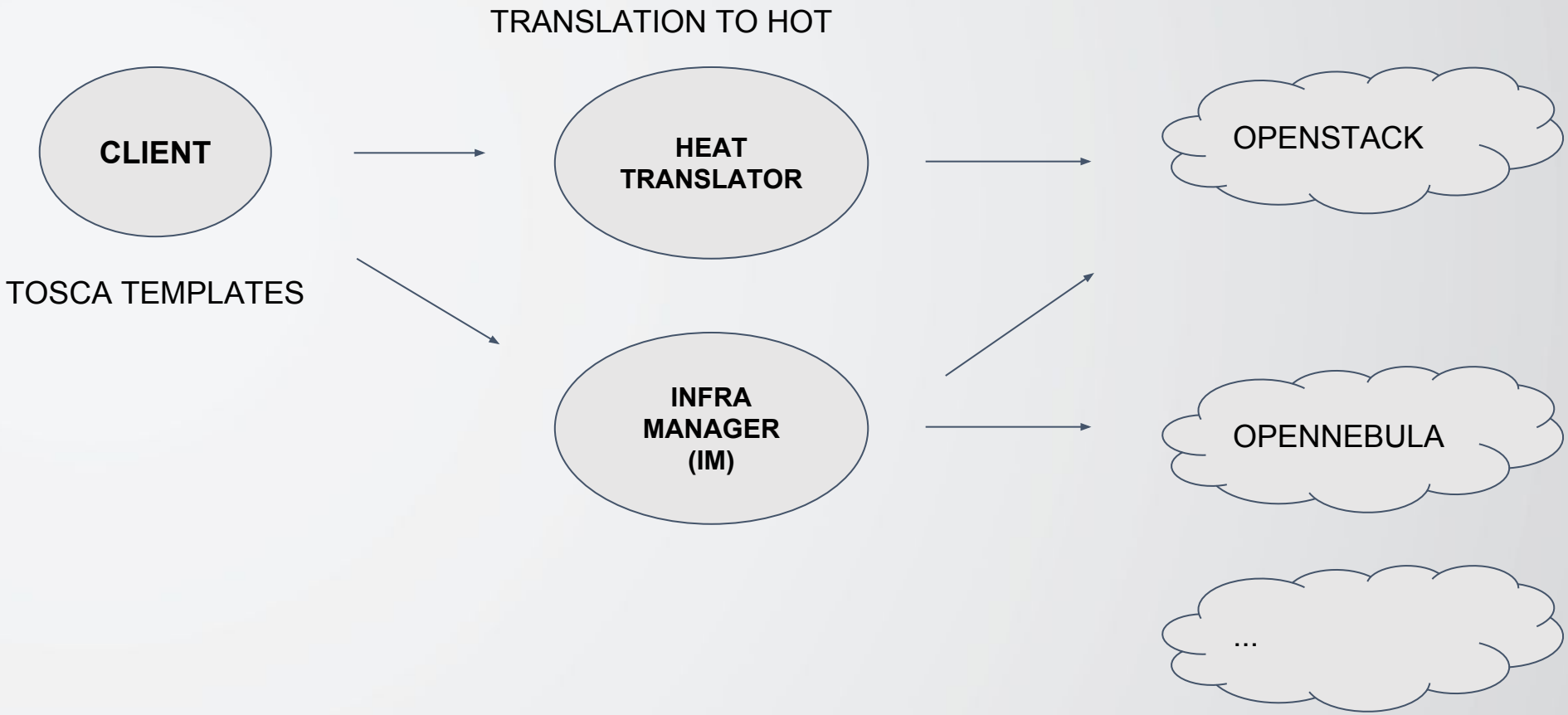
Why TOSCA?

- First step was to evaluate available orchestration options
  - HEAT, CloudFormation
  - Both tied to specific implementations
- TOSCA appeared as a viable common denominator
  - Topology definition
  - End user applications
- Existing codebase
  - TOSCA parser, HEAT translator
  - Reusable in other contexts
- Growing support in different communities

What is TOSCA?

- Topology and Orchestration Specification for Cloud Applications
  - OASIS standard
- Simple Profile for YAML v1.0
- Simple Profile for Network Function Virtualization (NFV)
- Goals
  - Portable deployment to any compliant cloud
  - Smoother migration of existing applications to the cloud
  - Flexible bursting (consumer choice)
  - Dynamic, multi-cloud provider applications

How it works


TRANSLATION TO HOT
CLIENT
TOSCA TEMPLATES
HEAT TRANSLATOR
INFRA MANAGER (IM)
OPENSTACK
OPENNEBULA
...

Basic Example: Indigo ComputeNode

```yaml
tosca_definitions_version: tosca_simple_yaml_1_0

imports:
 - indigo_custom_types: custom_types.yaml

description: >
 TOSCA test for launching compute node with a specified image and
getting
 as an output the IP and SSH credentials to access
topology_template:
 node_templates:
      ------------------------------------------->

 outputs:
   node_ip:
     value: { get_attribute: [ simple_node, public_address, 0 ] }
```

```yaml
simple_node:
  type: tosca.nodes.indigo.Compute
  properties:
    public_ip: yes
  capabilities:
    scalable:
      properties:
        count: 1
    host:
      properties:
        num_cpus: 1
        mem_size: 1 GB
    os:
      properties:
        type: linux
        distribution: ubuntu
        version: 12.04
```

Basic Example: Indigo ComputeNode

```
tosca.nodes.indigo.Compute:
  derived_from: tosca.nodes.indigo.MonitoredCompute

tosca.nodes.indigo.MonitoredCompute:
  derived_from: tosca.nodes.Compute
  properties:
    zabbix_server:
      type: string
      required: no
      default: 90.147.170.165
    zabbix_server_port:
      type: PortDef
      required: no
      default: 32314
    zabbix_server_metadata:
      type: string
      required: no
      default: Linux 668c875e-9a39-4d...
```

```
interfaces:
  Standard:
    create:
      implementation: zabbix_agent_install.yml
    configure:
      implementation: zabbix_agent_configure.yml
      inputs:
        zabbix_server: { get_property: [ SELF, zabbix_server ] }
        zabbix_server_port: { get_property: [ SELF, zabbix_server_port ] }
        zabbix_server_metadata: { get_property: [ SELF, zabbix_server_metadata ] }
    start:
      implementation: zabbix_agent_start.yml
```

Basic Example: HOT

```
heat_template_version: 2013-05-23
description: >
 TOSCA test for launching compute node with a specified
 image and getting as an output the IP and
 SSH credentials to access
parameters: {}
resources:
  ------------------------------------------->
outputs:
 node_ip:
   description: null
   value:
     get_attr:
     - simple_node
     - networks
     - private
     - 0
```

```
 simple_node:
   type: OS::Nova::Server
   properties:
     flavor: m1.small
     image: ubuntu-12.04-software-config-os-init
     public_ip: true
     user_data_format: SOFTWARE_CONFIG
 simple_node_create_config:
...
 simple_node_create_deploy:
...
 simple_node_configure_config:
   type: OS::Heat::SoftwareConfig
   properties:
     config:
       get_file: zabbix_agent_configure.yml
     group: ansible
 simple_node_configure_deploy:
   type: OS::Heat::SoftwareDeployment
   properties:
     config:
       get_resource: simple_node_configure_config
     input_values:
       zabbix_server: 90.147.170.165
       zabbix_server_metadata: Linux      668c875e-9a39-4d...
       zabbix_server_port: 32314
     server:
       get_resource: simple_node
```

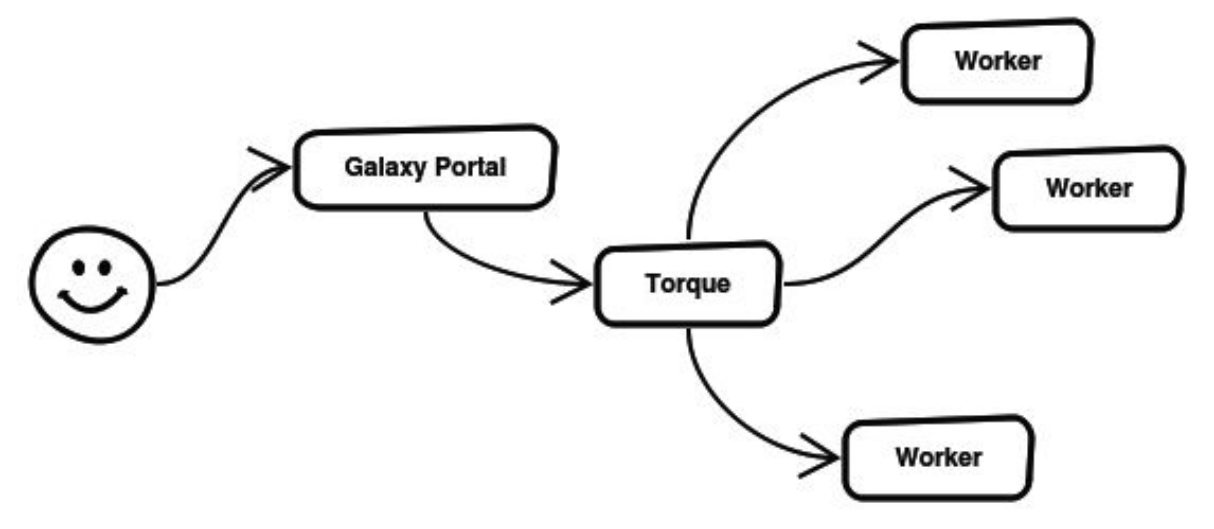
INDIGO - DataCloud

Multiple deployment & config options

- Supported in the HEAT Translator
  - .yaml is Ansible
  - .pp is puppet
  - .(anything else) is scripts (python, bash, …)
- This is reflected on the resulting HOT templates
- Extending the translator with others should not be hard
  - We did it for Ansible

Complex example: Batch Cluster

- A user facing portal manages jobs running on a batch cluster
- The cluster may spawn multiple sites
- Sites may be running different cloud flavors
  - Mainly OpenStack and OpenNebula



Complex example: Batch Cluster

```yaml
node_templates:

  elastic_cluster_front_end:
    type: tosca.nodes.indigo.ElasticCluster
    properties:
      deployment_id: orchestrator_deployment_id
    requirements:
      - lrms: torque_front_end
      - wn: wn_node

  torque_front_end:
    type: tosca.nodes.indigo.LRMS.FrontEnd.Torque
    requirements:
      - host: torque_server

  torque_server:
    type: tosca.nodes.indigo.Compute
    properties:
      public_ip: yes
    capabilities:
      host:
        properties:
          num_cpus: 1
          mem_size: 1 GB

  wn_node:
    type: tosca.nodes.indigo.LRMS.WorkerNode.Torque
    capabilities:
      wn:
        properties:
          max_instances: 5
          min_instances: 0
    requirements:
      - host: torque_wn

  torque_wn:
    type: tosca.nodes.indigo.Compute
    properties:
      public_ip: no
    capabilities:
      scalable:
        properties:
          count: 0
      host:
        properties:
          num_cpus: 1
          mem_size: 1 GB
      os:
        properties:
          type: linux
          distribution: ubuntu
          version: 14.04
```

Conclusions

- Many use cases already covered / implemented in TOSCA
  - Galaxy portal / SLURM
  - Mesos
  - DisVis
- Significant contributions upstream
  - Miguel core dev of the OpenStack TOSCA parser (and 2nd top contributor)
  - Mathieu core dev of the OpenStack Heat Translator (and 2nd top contributor)
- Available in the Indigo DataCloud Release 1
- Integration with the common openstack client