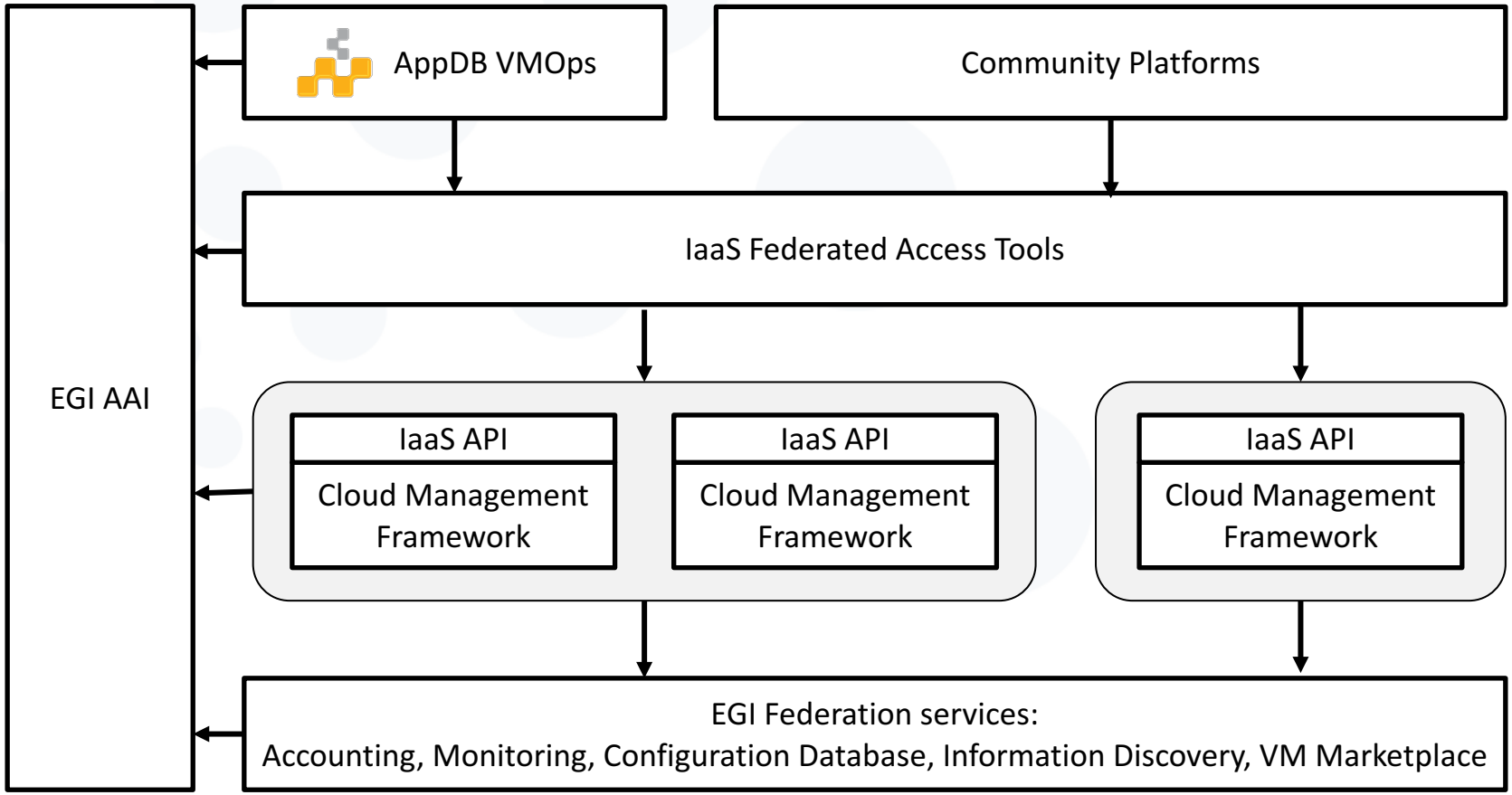
IaaS Orchestration on EGI Federated Cloud

Enol Fernández

Cloud Technologist @ EGI Foundation

Architecture: IaaS Federation



IaaS Federated Access Tools

- Provide access to the heterogeneous IaaS frameworks:
  - **IaaS provisioning systems that allow to define infrastructure as code** to manage and combine resources from different providers, thus enabling the portability of application deployments between them
  - *Smart* brokers providing matchmaking for workloads to available providers
  - Cloud Management Software that provides a unified console for accessing resources and deploy workloads following a set of user-defined established policies (e.g. Scalr or RightScale)

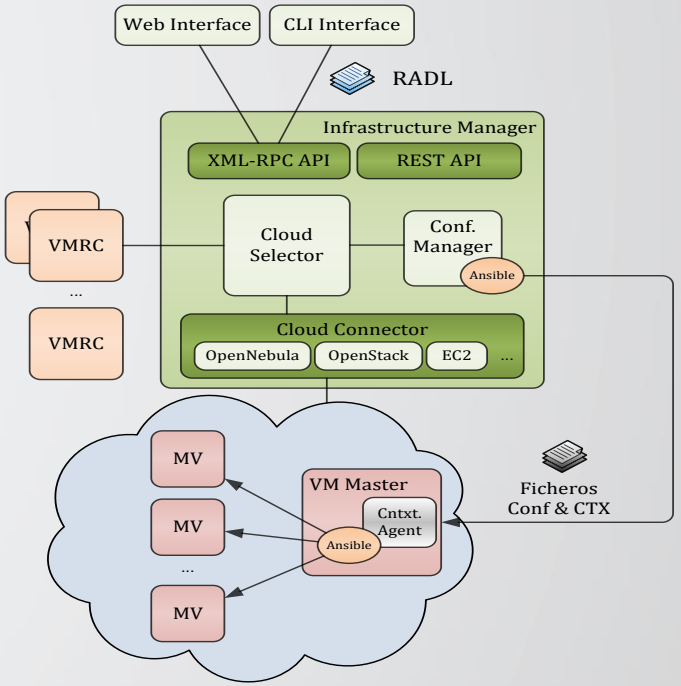
Evaluated tools

Tool	Supported EGI Cloud Interfaces	Infrastructure description	Deployment	Web GUI	CLI
IM	OCCI, OpenStack	RADL/**TOSCA**	Server	Yes	Yes
Terraform	OCCI, OpenStack	Terraform configurations	Client-side tool	No	Yes
OCCOPUS	OCCI, OpenStack	Occopus infrastructure description	Client or server	No	Yes
SlipStream	OCCI*	SlipStream Applications	Server	Yes	Yes

IM – Slides from INDIGO-DataCloud

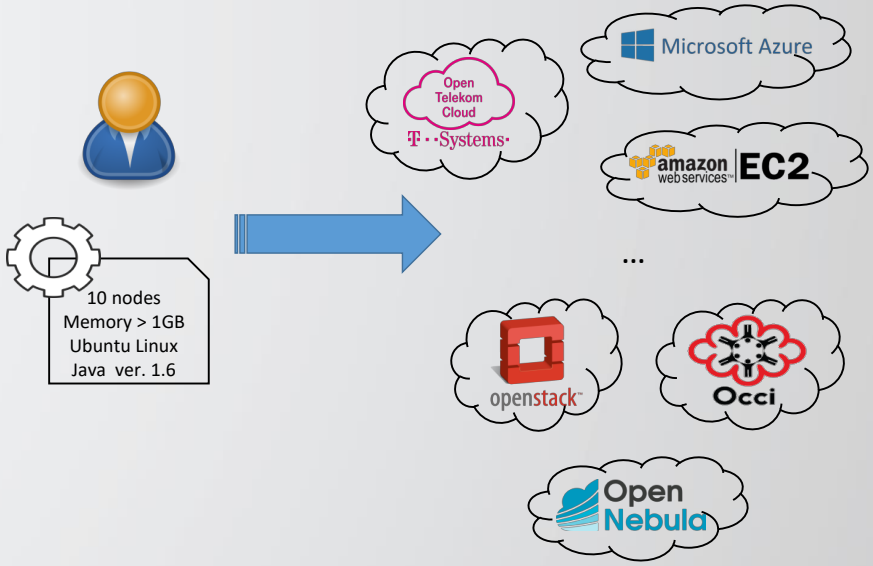
Introduction

- General platform to deploy on-demand customized virtual computing infrastructures.
  - With the precise software configuration required.
  - Complex infrastructures.
  - Share Infrastructure descriptions.
  - No need of pre-baked VMIs.
  - The same complex infrastructure can be deployed both on on-premises and on public Clouds.



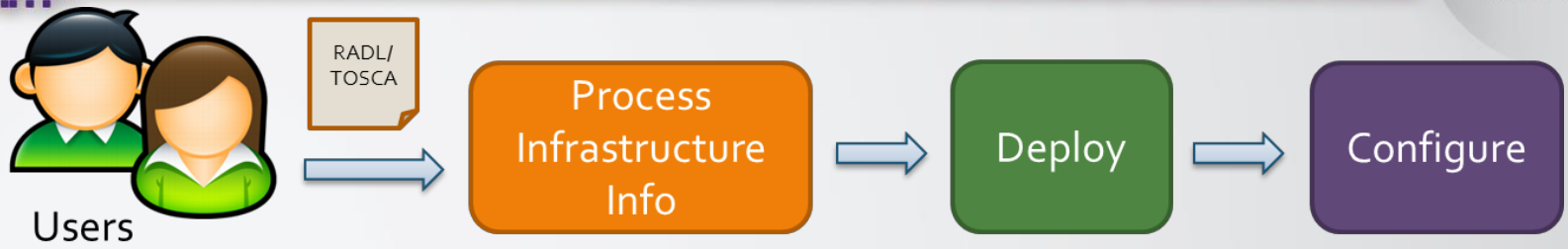
Cloud providers

- It supports a wide range of cloud providers and other computing back-ends :
  - Public: Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, Open Telekom Cloud (OTC).
  - On-premises: OpenNebula, OpenStack.
  - Federated: EGI FedCloud (OCCI), FogBow.
  - Containers: Docker, Kubernetes
  - The list above can be easily extended by plugins.



IM – Slides from INDIGO-DataCloud

Infrastructure Manager



- The user can provide an RADL or TOSCA documents as input to the IM, describing the infrastructure:
  - **RADL**:
    - Resource and Application Description Language.
    - High level Language to define virtual infrastructures and Specify VM requirements.
  - **TOSCA**:
    - OASIS Standard
    - Open standard language to model application architectures to be deployed on a Cloud.

RADL Document

- An RADL document has the following general structure:

```
ansible <ansible_host_id> (<features>)

network <network_id> (<features>)

system <system_id> (<features>)

configure <configure_id> (<Ansible recipes>)

contextualize [max_time] ( system <system_id>

configure <configure_id> [step <num>] ... )

deploy <system_id> <num> [<cloud_id>]
```

The keywords ansible, network, system and configure assign some *features* or *recipes* to an identity <id>. The features are a list of constraints separated by and, and a constraint is formed by <feature name> <operator> <value>.

```
network net (outbound = 'yes')

system node (
  cpu.arch = 'x86_64' and  cpu.count = 1 and
  memory.size >= 512M and
  net_interface.0.connection = 'net' and
  disk.0.os.name = 'linux' and
  disk.0.image.url = 'one://onecloud.i3m.upv.es/67'
)

configure node (
@begin
 - tasks:
   - user: name=user1 password=1234
 @end
)

contextualize (
  system node configure node
)

deploy node 1
```

IM – Slides from INDIGO-DataCloud

A Sample tosca template: kepler

```
tosca_definitions_version: tosca_simple_yaml_1_0
imports:
 - indigo_custom_types:  custom_types.yaml
topology_template:
 node_templates:
   kepler:
    type: tosca.nodes.indigo.Kepler
    requirements:
     - host: kepler_server
   kepler_server:
    type: tosca.nodes.indigo.Compute
    capabilities:
     host:
      properties:
       num_cpus: 1
       mem_size: 1 GB
      endpoint:
       properties:
        network_name: PUBLIC
        ports:
         vnc_port:
          protocol: tcp
          source: 5900
     os:
      properties:
       type: linux
       distribution: ubuntu
       version: 14.04
       image: one://onecloud.i3m.upv.es/67

outputs:
 instance_ip:
  value: { get_attribute: [ kepler_server, public_address, 0 ] }
 instance_creds:
  value: { get_attribute: [ kepler_server, endpoint, credential, 0 ] }
```

Network requirements

OS requirements

Terraform

- Terraform is a tool for building, changing, and versioning infrastructure safely and efficiently.

- Goals:
  - Unified view of infrastructure
  - Infrastructure as Code
  - Compose multiple tiers (IaaS, PaaS, SaaS)
  - Safely change infrastructure over time
  - One workflow

HashiCorp
Terraform

Terraform Key Features

- Infrastructure as Code
  - Declare using a high-level configuration syntax
- Resource Providers
  - IaaS (e.g. AWS, GCP, Microsoft Azure, OpenStack)
  - PaaS (e.g. Heroku)
  - SaaS services (e.g. DNSimple, CloudFlare)
- Execution Plans
  - Avoid surprises when manipulating the infrastructure
- Resource Graph
  - Parallelizes the creation and modification of any non-dependent resources

An example

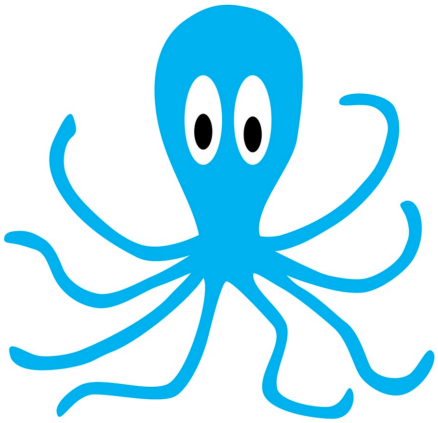
```
resource "openstack_compute_keypair_v2" "test-key" {
  name = "my-keypair"
  public_key = "${file("~/.ssh/id_rsa.pub")}”
}

resource "openstack_compute_floatingip_v2" "floatip_1" {
  pool = "provider”
}

resource "openstack_compute_instance_v2" "master" {
  name = "master”
  image_id = "befecd08-78c2-4177-bbf5-4afd462f5d09"
  flavor_id = "308bc2b2-1e1e-4af9-a98f-cac76b6ce084"
  key_pair = "${openstack_compute_keypair_v2.test-key.name}”
  security_groups = ["default"]
  network {
    floating_ip = "${openstack_compute_floatingip_v2.floatip_1.address}"
    access_network = true
  }
}
```

Terraform + FedCloud

- Good support for different kind of resources and support for combining them together
  - Providers developed by EGI to incorporate support (OpenStack/OCCI)
- No central server, no external dependencies, just one binary
- Configurations are not that easy to port between different types of providers :(

Occopus

- Occopus is a hybrid cloud orchestration tool
  - Configures “Virtual Infrastructures” on single or multi cloud



- Key Features:
  - Command line tool and REST API service
  - Multi-cloud support
  - Pluggable architecture
  - Error-detection (fatal/transient) and recovery
  - Support for configuration management tools (like Chef)
  - Garbage collection at VM cancellation
  - Manual scaling

Occopus: defining infrastructures

- Virtual infrastructure description:
  - Nodes (services) to be deployed and all cloud independent attributes (e.g. input values for a service)
  - Dependencies among the nodes, to decide the order of deployment
  - Scaling related attributes like min, max number of instances
- Node definition:
  - how to construct the node on a target cloud, i.e. all cloud dependent settings, e.g. image id, flavour, contextualization

Example: infrastructure description

```
nodes:
    - &DBS_Node
        name: mysql_server
        type: ec2_chef_mysql_server_node
    - &WP_Node
        name: wordpress
        type: ec2_chef_wordpress_node

dependencies:
    -
        connection: [ *WP_Node, *DBS_Node ]
```

Example: node definition

```
'node_def:ec2_chef_mysql_server_node':
    -
        resource:
            type: occi
            endpoint: https://carach5.ics.muni.cz:11443
            os_tpl: os_tpl#uuid_egi_ubuntu_server_14_04_lts_fedcloud_warg_131
            resource_tpl: http://fedcloud.egi.eu/occi/compute/flavour/1.0#medium
            link:
                -
                    https://carach5.ics.muni.cz:11443/network/24
        contextualisation:
            type: cloudinit
            context_template: !text_import
                url: file://cloud_init_wordpress.yaml
            attributes:
                mysql:
                    server_root_password: '{{ variables.mysql_root_password }}'
        config_management:
            type: chef
            endpoint: https://c155-14.localcloud
            run_list:
                - recipe[database-setup::db]
        health_check:
            mysqldbs:
                - {name: my_DB,
                   user: my_user,
                   pass: '{{ variables.mysql_dbuser_password }}'}
```

More information

Check:

https://wiki.egi.eu/wiki/Federated_Cloud_IaaS_Orchestration

Or email enol.fernandez@egi.eu for support

Thank you for your attention.

Questions?

www.egi.eu