Scaling Science Clouds in Europe with TOSCA and Heat Translator

Mathieu Velten
mathieu.velten@cern.ch
@MatMaul

Sahdev P. Zala
spzala@us.ibm.com
@sp_zala

Alvaro Lopez Garcia
aloga@ifca.unican.es
@alvaretas

Miguel Caballer
micafer1@upv.es
@micafer77
Introduction

- In this presentation we will cover:
  - How we are handling interoperability challenge across heterogeneous clouds
  - TOSCA overview and latest development
  - TOSCA Parser and Heat Translator overview and latest development
  - Project INDIGO-DataCloud deep drive and latest development
  - INDIGO-DataCloud use case to model full stack with TOSCA

- INDIGO-DataCloud Project:
  - Large scale project to provide a sustainable cloud infrastructure for scientific community

- Challenges:
  - Need to effectively exploit heterogeneous cloud infrastructures (OpenStack, OpenNebula, proprietary clouds) as a single cloud
  - Need a standard orchestration model that is interoperable across multiple cloud infrastructures

- Our Approach:
  - Leverage TOSCA as a standard based approach for modeling cloud stacks and applications
  - Leverage OpenStack components as a means of deploying TOSCA cloud stacks in OpenStack cloud
    - TOSCA Parser library
    - Heat Translator CLI tool
    - Heat orchestration service
TOSCA and OpenStack
TOSCA - Topology and Orchestration Specification for Cloud Applications

- An important Open standard, that is enabling a unique interoperable Cloud eco-system supported by a large and growing number of international industry leaders...

TOSCA uses a domain-specific language (DSL) to define interoperable descriptions of:

- Cloud applications, services, platforms, infrastructure and data components, along with their relationships, requirements, capabilities, configurations and operational policies...

- ...thereby enabling portability and automated management across cloud providers regardless of underlying platform or infrastructure thus expanding customer choice, improving reliability and time-to-value while reducing costs.
TOSCA Spec: Important Milestones and Contributors

- **TOSCA Version 1.0 Specification approved as an OASIS Standard**
  - Published Nov 2013, XML format
- **TOSCA Simple Profile Specification in YAML**
  - **V1.0 Approved June 2016**
  - v1.1 under public review currently
- **TOSCA NFV Profile v1.0**
  - Stable draft available. WIP.
- **Government and Corporate Awareness:**
  - OASIS: 600+ participant organizations. 5000+ participants spanning 65+ countries
  - TOSCA Committee: 170+ people 45+ companies/orgs
  - International Standards & Research: ISO/IEC JTC 1 liaison, EU FP7, ETSI NFV liaison, etc.
- **Multi-company Interoperability Demonstrated:**
  - EuroCloud 2013, Open Data Center Alliance 2014, OSCON 2015, OpenStack Summit 2016 (Indigo DataCloud)

Associated Companies

Includes contributors, reviewers, implementers, users or supporters of the TOSCA Standard via OASIS
OpenStack Newton: TOSCA Parser

- What is the TOSCA Parser?
  - A parser for TOSCA Simple and Network Functions Virtualization (NFV) YAML based specifications
  - Produces in-memory graph of TOSCA nodes and relationship among them
  - Subproject of OpenStack Heat main project
- Two point releases – 0.5.0 and 0.6.0
  - 0.5.0
    - Release date: May 04, 2016
    - Parsing support for Policies and Triggers, and LoadBalancer type
    - Support for complex types like TOSCA PortSpec, bug fixes etc.
  - 0.6.0
    - Release date: August 08, 2016
    - Python 3.5 support with new gate job for 3.5 enabled
    - Support for TOSCA Repositories which are an external repositories in TOSCA service template containing deployment artifacts
    - NFV specific updates per latest development in the NFV specification
    - Backward compatibility, bug fixes etc.
- PyPi releases are available at [https://pypi.python.org/pypi/tosca-parser](https://pypi.python.org/pypi/tosca-parser)
OpenStack Newton: Heat Translator

- What is the Heat Translator?
  - An OpenStack project to translate non-Heat (e.g. TOSCA) templates to HOT, and deploy it with Heat
  - Subproject of OpenStack Heat main project

- Two point releases – 0.5.0 and 0.6.0

  0.5.0
  - Release date: May 13, 2016
  - Support for Ansible and Puppet scripts in translation
  - Dynamic image querying with Glance for TOSCA operating system constraints (distro, version etc.)
  - Automatic deployment of translated template
  - Backward compatibility with new gate job for TOSCA Parser

  0.6.0
  - Release date: Sep 22, 2016
  - Python 3.5 support with new gate job for 3.5 enabled
  - Automatic deployment with proper authentication using Keystone (i.e. KeystoneAuth Library) vs relying on environmental variables OS_* (e.g. OS_USER)
  - Translation support for AutoScalingGroup, ScalingPolicy and Aodh resources with multiple translated templates
  - Translation support for Senlin cluster and auto scaling policy resources
  - Handling of Ansible roles used with TOSCA artifacts

- WIP: Heat Translator as a Service to be easily consumed by adopters like project INDIGO-DataCloud, Tacker etc.

- PyPi releases are available at, https://pypi.python.org/pypi/heat-translator
INDIGO-DataCloud overview
INDIGO-DataCloud Project

- Towards a sustainable European IaaS, PaaS and SaaS for computing and data, focused on science and scientific users
- Funded under the European Commission H2020 program (grant agreement RIA 653549), started in 2015
- 26 partners, 11 countries:
  - INFN, CSIC, CERN, DESY, Atos, UPV, LIP, T-Systems, CNRS, Reply, etc.
- Heterogeneous infrastructure (OpenStack, OpenNebula, commercial providers)
- Gathering requirements from 11 different, multidisciplinary communities
  - LifeWatch, EuroBioImaging, INSTRUCT, LBT, CTA, WeNMR, ENES, eCulture Science Gateway, ELIXIR, EMSO, DARIAH, WLCG.

https://www.indigo-datacloud.eu
INDIGO-DataCloud Foundations

- **Users first**: involve researchers, scientific resource centers, industry, software developers
- Define and validate software components to be developed through concrete scientific use cases
- Develop open source software filling technological gaps that hinder the exploitation of current e-infrastructures for science
  - Reuse and extend existing components wherever possible
  - Develop missing pieces whenever necessary
  - Exploit 15 years of previous experience of partner participants
- Be as multidisciplinary as possible
- Be vendor neutral, through the adoption and promotion of standards, aiming at interoperability between clouds
INDIGO-DataCloud User Communities

- 11 different user communities
  - LifeWatch, EuroBioImaging, INSTRUCT, LBT, CTA, WeNMR, ENES, eCulture Science Gateway, ELIXIR, EMSO, DARIAH, WLCG.
- Distributed all over Europe
- Several ESFRIs involved (in red)
INDIGO-DataCloud and TOSCA
Why do we need TOSCA anyway?

- Orchestration is needed to manage complex interactions and workflows
  - An interoperable heterogeneous infrastructure is a key issue (OpenStack, OpenNebula)
- First step, evaluation of available 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)
  - Separated parser and translator components, easy to consume
  - Reusable in other contexts too
- Growing support in different communities, backed by industry (like IBM)
How INDIGO-DataCloud Leverages TOSCA?

- We define custom types covering our use cases
  - Extending the TOSCA Simple Profile in YAML
  - [https://github.com/indigo-dc/tosca-types](https://github.com/indigo-dc/tosca-types)

Some examples:

- Elastic cluster for batch processing (infrastructure)
  - Supported systems: Slurm (official), Torque, HTCondor
- Apache Mesos cluster deployment (infrastructure)
- Galaxy portal (end user)
  - Data intensive application for biomedical research
- INDIGO-Datacloud specific jobs (infrastructure / end user)
  - Package and deploy applications in Docker containers, on Mesos/Marathon/Chronos
Indigo Integrators: Create TOSCA templates for their application stack
Dashboard: Select and parameterize an application stack
Multi sites orchestrator: TOSCA templates with parameters
Infrastructure Manager (from UPV): Orchestration
OpenStack Heat Translator
OpenStack Heat Base Services
OpenStack Heat
GCE, AWS, OpenNebula: Connectors
OpenNebula Base Services
IaaS
WIP on the translator

- Better handling of scalable resources
  - Generic framework, can be mapped to ResourceGroup / AutoScalingGroup / Senlin Cluster
  - Support software deployments, dependencies, parameters passing
  - Use a substack for each scalable resource

- Handling of capabilities. Endpoint on compute node
  - Translate to a network directive on the generated Nova server descriptor and SecurityGroup if port(s) specified
Scalable compute node with SW deployment

inputs:
  install_path: type: string
node_templates:
  scalable_node:
    type: tosca.nodes.Compute
capabilities:
  scalable:
    properties:
      count: 2
interfaces:
  Standard:
    create:
      inputs:
        install_path:
          get_parameter: install_path
      implementation: install.yml
parameters:
  install_path: type: string
resources:
  scalable_node:
    type: OS::Nova::Server
  scalable_node_create_deploy:
    type: OS::Heat::SoftwareDeployment
    properties:
      input_values:
        install_path:
          get_param: install_path
      config:
        get_resource: scalable_node_create_config
      server:
        get_resource: scalable_node
  scalable_node_create_config:
    type: OS::Heat::SoftwareConfig
    properties:
      group: ansible
      config:
        get_file: install.yml
Network endpoint support

Call neutron to know public network name

```
apache_node:
  type: tosca.nodes.Compute
  properties:
    network_name: PUBLIC
    ports:
      http:
        protocol: tcp
        source: 80

internal_node:
  type: tosca.nodes.Compute
  properties:
    network_name: myprivatenetwork

apache_node:
  type: OS::Nova::Server
  properties:
    networks:
      - network: CERN_NETWORK
    security_groups:
      - get_resource: sec_group
  sec_group:
    type: OS::Neutron::SecurityGroup
    properties:
      rules:
        - remote_ip_prefix: 0.0.0.0/0
          protocol: tcp
          port_range_min: 80
          port_range_max: 80

internal_node:
  type: OS::Nova::Server
  properties:
    networks:
      - network: myprivatenetwork
```
Multiple deployment & config options

- Supported in the HEAT Translator
  - .yaml is Ansible
  - .pp is puppet
  - .(anything else) is scripts (python, bash, ...)

- Galaxy Ansible roles defined as artifacts are automatically installed
  - Can then be used in the Ansible scripts

```yaml
node:
  type: tosca.nodes.Compute
artifacts:
  zabbix_role:
    type: tosca.artifacts.AnsibleGalaxy.role
    file: indigo-dc.zabbix-agent
interfaces:
  Standard:
    create: install_zabbix.yml
```
Conclusions
INDIGO-DataCloud TOSCA Status

- First use cases already covered by existing tools
- INDIGO-DataCloud contributions upstream
  - TOSCA Parser
    - UPV 2nd contributor in number of commits (after IBM :-))
  - Heat Translator
    - CERN 2nd contributor in number of commits (after IBM :-))
- Using the Heat Translator CLI, the common openstack client, and in some cases other APIs
- Currently supports several IaaS, but only deploy a full cluster in one. Final goal is to launch one across several clouds
What’s next?

- Define additional TOSCA custom types for the remaining INDIGO-DataCloud use cases
  - Ensure that support in the Heat Translator is in place
- Expand deployment to more sites within INDIGO-DataCloud
- Continue collaboration between INDIGO-DataCloud and upstream devs
- Heat Translator as a Service endpoint, able to consume TOSCA directly
Thank You!

Special credits to: Miguel Caballer (UPV), Germán Moltó (UPV), Marica Antonacci (INNF), Giacinto Donvito (INFN), Ricardo Rocha (CERN), INDIGO-DataCloud, and upstream TOSCA Parser and Heat Translator contributors...