ONAP Model Driven Orchestration with TOSCA

Arthur Berezin, ONAP TSC Member, Cloudify
AGENDA

• Why TOSCA is one of ONAP’s main Pillars?
• The Model driven Vision
• How TOSCA Works?
• TOSCA in ONAP
  • OPNA R1(Amsterdam) and R2(Beijing)
• TOSCA Hands On Lab with Apache ARIA-TOSCA
Linux Foundation Framework, Governance, Control
Bringing the best of both worlds together

+ 2+ years of Deployment Maturity at AT&T
+ Comprehensive: Design +Orchestration + Control + Policy + Analytics
+ Model-based design enabling self-serve capabilities for instantiation and closed loop automation

+ Open TOSCA model
+ Most Advanced Open Source Process & tool chain
+ Architected for ease of VNF insertion (SDK)
From OPEN-O: open TOSCA model

**Design-time**
- Portal
- OPEN-O UI (GUI/CLI)
- OPEN-O UI (GUI/CLI)
- SDC
- VNF Design
- Service Design
- Workflow Design
- Policy Creation
- Analytic Application Creation
- Recipie/Engineering Rules & Policy Distribution
- Catalog
- VNF SDK

**Run-time**
- Dashboard
- Service Orchestration
- External Data Movement & APIs
- A&AI
- DMaaP
- ESR
- Auth.
- Microservice Bus
- Common Service

**Controllers**
- Controllers
- Infra-C
- SDN-C
- APP-C
- SDN Hub Driver
- SDN Agent (SDN-O)
- Multi-VIM
- Multi VNFM/EM Driver

**Cloud & WAN**
- OpenStack
- VMware
- RackSpace
- Azure
- ......
ONAP Amsterdam Use Cases

vCPE

vVoLTE
HARD CODING
Standard Modeling is Automation at Scale

Example from the Manufacturing Industry

The Boeing 787 could not be produced without standard modeling.
Declarative Model Driven Orchestration

ONAP TOSCA Service Orchestrator
(SO+Apache ARIA)

Images
Scripts
Artifacts

Any Cloud
Config MGT
Networking
Containers

Service Description
Service Instantiation
Declarative Model Driven Orchestration

- Declarative Orchestration separates the **WHAT** from the **HOW**
- The declarative defines a model (what) and an orchestrator (how)
- Model provides info for orchestrator to remain generic
- A Model Driven Approach allows:
  - The allure of no coding
  - Orchestrator implementation hidden
  - Simple versionable artifact(s)
  - Designer focus on value/solution
TOSCA Basics - Node Types

- Node Types > Node Templates
- Nodes are associated with
  - Properties(Static)
  - Attributes/Runtime)
  - Life-cycle Interfaces
    - Lifecycle operations and implementations (create, configure, start and others)
- Normative node types (Compute, Network, etc’)
- Direct Relationship types
  - Connected-to, Hosted-on
TOSCA Basics - Application Template

- TOSCA Template YAML file describes the topology of the application.
- Normative and custom types can be used
- Topology templates use TOSCA node types describing the nodes and their relationships
  - Containment Relationship
  - Connection Relationship
  - Requirements and Capabilities
- Templates also define implementations for lifecycle operations.
Model Driven Orchestration

tosca_app_template.yaml

tosca_types.yaml

Deployment STATE

INSTALL Workflow Execution
TOSCA Hello World

Node Types

WebServer:
- derived_from: tosca:Root
- capabilities:
  - host:
    type: tosca:Container
WebApp:
- derived_from: tosca:WebApplication
- properties:
  - port:
    type: integer

Properties

WebApp:
- properties:
  - port:
    type: integer

Requirements

- host: web_server

Lifecycle Interfaces

configure:
- scripts/configure.sh

start:
- scripts/start.sh

stop:
- scripts/stop.sh

Outputs

- outputs:
  - port:
    type: integer
    value: { get_property: [ web_app, port ] }
TOSCA In ONAP

Service Orchestration (SO)

VID

ONAP Service Package

BPMN

TOSCA Engine

Multi-VIM

SDN-C

App-C

VF-C

DCAE

A&AI

TOSCA Template

Images

Scripts

Artifacts

Service Design and Creation (SDC)
Requirements and R1 Contributions

- ARIA must be callable from Camunda and/or seed code
  - Contribution: REST API [https://gerrit.onap.org/r/gitweb?p=so.git;a=tree;f=aria/aria-rest-server](https://gerrit.onap.org/r/gitweb?p=so.git;a=tree;f=aria/aria-rest-server)
  - Contribution: ARIA Java Binding [https://gerrit.onap.org/r/gitweb?p=so.git;a=tree;f=aria/aria-rest-java-client](https://gerrit.onap.org/r/gitweb?p=so.git;a=tree;f=aria/aria-rest-java-client)

- ARIA needs to fit in ONAP deployment model & be a microservice
  - Contribution: ARIA Docker Image [https://gerrit.onap.org/r/gitweb?p=so.git;a=tree;f=aria/multivim-plugin](https://gerrit.onap.org/r/gitweb?p=so.git;a=tree;f=aria/multivim-plugin)

- ARIA needs to communicate with MultiVIM service
  - Contribution: ARIA ONAP MultiVIM plugin [https://gerrit.onap.org/r/gitweb?p=so.git;a=tree;f=aria/multivim-plugin](https://gerrit.onap.org/r/gitweb?p=so.git;a=tree;f=aria/multivim-plugin)

- vCPE use case
  - TOSCA VNF vCPE infrastructure template (HEAT equiv) [https://gerrit.onap.org/r/gitweb?p=demo.git;a=tree;f=tosca/vCPE/infra/base_vcp_einfra_rackspace_tosca.yaml](https://gerrit.onap.org/r/gitweb?p=demo.git;a=tree;f=tosca/vCPE/infra/base_vcp_einfra_rackspace_tosca.yaml)
Full Integration With Core BPMN

• Approach: Post SDNC/OF fork
• Implications:
  - ONAP Base types for AAI, MultiVIM, APPC
  - Python APPC/DMaaP API
  - SDC ONAP Base type integration (potentially)
  - SDNC & OF ONAP types recognition/compatibility
  - SO/ARIA BPMN recognition and handoff (via inputs)
    • includes mapping tenant info to ARIA template names & processing outputs
  - ONAP aware workflows (at least install/uninstall)
  - DCAE init ( BPMN or TOSCA workflow )
  - Enhance/complete ARIA REST API, improve microservice impl
• Minimum proof of work: pure TOSCA vCPE install/uninstall
Apache ARIA-TOSCA

Fortinet Fortigate Firewall
https://github.com/dfilppi/fortigate-tosca-example

Fortigate Image(DEMO Purpose)
https://s3-eu-west-1.amazonaws.com/cloudify-labs/images/FG562-DZ.img

Env:

VNF TOSCA Model Generator(Alpha)
http://185.43.218.204
1. SSH Into ARIA Machine
   $ssh cloudify@185.98.149.13 (password:'#######')
2. Sudo into root
3. $ sudo su - (password again)
4. Run your ARIA-Workshop Docker container:
5. # docker run -ti --rm --name <MY-COMP> aria-workshop /bin/bash
aria service-templates store fortigate-vnf-baseline-tosca.yaml
<SERVICE-NAME>
aria services create -t <TEMPLATE>
aria executions start -s <SERVICE> install
ARIA-TOSCA Examples and Resources

- Hello World
  - [https://github.com/apache/incubator-ariatosca/tree/master/examples/hello-world](https://github.com/apache/incubator-ariatosca/tree/master/examples/hello-world)
- ClearWater IMS (On pre installed VM)
  - [https://github.com/apache/incubator-ariatosca/tree/master/examples/clearwater](https://github.com/apache/incubator-ariatosca/tree/master/examples/clearwater)
- Fortinet Fortigate Firewall (OpenStack)
  - [https://github.com/dfilppi/fortigate-tosca-example](https://github.com/dfilppi/fortigate-tosca-example)
- TOSCA 1.0 Simple Profile SPEC Use Case Implementations
  - [https://github.com/apache/incubator-ariatosca/tree/master/examples/tosca-simple-1.0/use-cases](https://github.com/apache/incubator-ariatosca/tree/master/examples/tosca-simple-1.0/use-cases)
TOSCA VNF Model Generator Wizard

http://185.43.218.204/#!/w/vnf
Apache ARIA-TOSCA Workshop

http://tinyurl.com/tosca-workshop
Thanks
Arthur@Cloudify.co
FROM centos
RUN yum install -y gcc python-devel epel-release && yum install -y python-pip
RUN pip install apache-ariatosca[ssh]
CMD aria -h
ARIA Workshop Dockerfile

```
FROM centos
RUN yum install -y gcc git python-devel epel-release && yum install -y python-pip
RUN pip install apache-ariatosca[ssh] wagon
RUN git clone https://github.com/cloudify-cosmo/aria-extension-cloudify.git /root/aria-extension-cloudify
RUN pip install -r /root/aria-extension-cloudify/requirements.txt
RUN pip install /root/aria-extension-cloudify/
RUN git clone -b 2.0.1 https://github.com/cloudify-cosmo/cloudify-openstack-plugin /root/cloudify-openstack-plugin
RUN wagon create /root/cloudify-openstack-plugin
RUN aria plugins install cloudify_openstack_plugin-2.0.1-py27-none-linux_x86_64.wgn
RUN git clone -b 1.3.0 https://github.com/cloudify-incubator/cloudify-utilities-plugin /root/cloudify-utilities-plugin
RUN wagon create /root/cloudify-utilities-plugin
RUN aria plugins install cloudify_utilities_plugin-1.3.0-py27-none-linux_x86_64.wgn
RUN git clone https://github.com/rtpro/fortigate-tosca-example.git
CMD aria -h
```

```
aria-workshop]# cat Dockerfile
FROM centos
RUN yum install -y gcc git python-devel epel-release && yum install -y python-pip
RUN pip install apache-ariatosca[ssh] wagon
RUN git clone https://github.com/cloudify-cosmo/aria-extension-cloudify.git /root/aria-extension-cloudify
RUN pip install -r /root/aria-extension-cloudify/requirements.txt
RUN pip install /root/aria-extension-cloudify/
RUN git clone -b 2.0.1 https://github.com/cloudify-cosmo/cloudify-openstack-plugin /root/cloudify-openstack-plugin
RUN wagon create /root/cloudify-openstack-plugin
RUN aria plugins install cloudify_openstack_plugin-2.0.1-py27-none-linux_x86_64.wgn
RUN git clone -b 1.3.0 https://github.com/cloudify-incubator/cloudify-utilities-plugin /root/cloudify-utilities-plugin
RUN wagon create /root/cloudify-utilities-plugin
RUN aria plugins install cloudify_utilities_plugin-1.3.0-py27-none-linux_x86_64.wgn
RUN git clone https://github.com/rtpro/fortigate-tosca-example.git
CMD aria -h
```

```
aria-workshop]# docker build -t aria-workshop .
```
TOSCA Model Driven