Orange

Our ideal ONAP

Eric Debeau
Sylvain Desbureaux
A service provider vision

- Feedbacks from many ONAP use-cases experiences
- Feedbacks from operational constraints
- Anticipation for future networks (5G, IoT...)
ONAP: A powerful framework

- Define Policy: That governs service and resource behavior
- Define Analytics: That monitor policies
- Heal & Scale: Take action to meet service level
- Monitor Service: By listening to events; computing analytics based on data collection.

- Define Service: Based on Resource Model and needs
- Recommend Changes: Analyze behavior over time to identify changes needed in designs, policies, analytics or thresholds governing response
- Log Events: Actor(s) publish events to record changes made for the required conditions
- Distribute: Design templates and policies to various actors
ONAP Journey

Dublin

Amsterdam

Casablanca

Beijing
Focus on 4 main areas

<table>
<thead>
<tr>
<th>Design-time</th>
<th>Run-time</th>
<th>Operations</th>
<th>Usability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modeling tasks</td>
<td>Execution engines</td>
<td>ONAP operations</td>
<td>Simplicity to use ONAP</td>
</tr>
<tr>
<td>Model verification</td>
<td>Apply model with context values</td>
<td>Configuration</td>
<td>Ease to model, reuse</td>
</tr>
<tr>
<td>Artifacts developments</td>
<td>Configuration</td>
<td>Monitoring</td>
<td>API compliance to ease integration</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Flexibility to deploy</td>
<td></td>
</tr>
</tbody>
</table>
Design-Time objectives

Towards zero code

- Model as much as possible
- Model verification tools
- Code limitation to specific
  - Policy
  - Analytics
  - Controller behavior
  - External interactions

VNF/PNF
Service
Policy
Control Loop
Design-Time status

More modeling, less code

- A lot of capabilities within the Service, Design and Creation (SDC)
  - Policy, Artifacts

- Controller Design Studio, Service Orchestrator
  - Reusable Building Blocks

- Policy definition based on TOSCA
Design-Time challenges

- Complex service composition
  - Slices, “shared” services

- Capability to cover all network domains
  - Variety of devices, configurations, operations

- Bridge various models all together
  - Service model mainly based on TOSCA
  - Configuration model based on a variety of models/languages

- Covering future requirements
  - CNF, 5G, IoT, distributed clouds, AI
Run-Time objectives

A single reliable path

- Aligned with modeling capabilities
- Flexible, Robust, Fast
- A layered framework
- A single API to deploy&configure a service
- ONAP configuration by API
Run-Time status

Powerful engine, with many paths

- Rich framework
  - Layer approach
  - Take into account new features

- Many paths
  - Deploy a VNF
  - Implement a Policy
  - Collect data
Run-Time challenges

- Reduce the number of paths
  - Complex to test

- Execution speed
  - Near Real-Time Control Loop

- Complex services, multi-domain
  - Service Resolver
Operations

Objectives
- Toolkit to operate ONAP in a secure environment
- Flexibility to deploy
- Optimized footprint

Status
- Powerful installer (OOM) enabling many configurations
- Footprint still to be improved

Challenges
- A small ONAP easy to deploy to help adoption
Usability

Objectives
- Easy to use, to start
- Re-use existing models, artifacts
- Unique portal per domain
- Compliancy with SDO for integration

Status
- Documentation
- Strong momentum for using SDO API

Challenges
- Keep documentation & tutorials aligned with every release
- Write documentation for newbies to experts
ONAP: A powerful framework with no limit thanks to open source and collaboration
Thank you