Demystifying the ONOS NB API

Brian O'Connor
ON.Lab
March 15, 2016

#ONOSProject
ONOS provides a **flexible and extensible** API with **multiple layers of abstraction** for both **network programming** and **configuration**.
How do we get there?

- Intent
- Topology
- Config
- Flow
- Device

Your Own Service

NBI

#ONOSProject
## Flexibility

**Interface DeviceService**

All Superinterfaces:
- ListenerService<DeviceEvent, DeviceListener>

All Known Subinterfaces:
- DeviceAdminService

Service for interacting with the inventory of infrastructure devices.

### Method Summary

<table>
<thead>
<tr>
<th>Modifier and Type</th>
<th>Method Call</th>
<th>Method and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>public</td>
<td>getAvailableDevices()</td>
<td>Returns an iterable collection of all devices currently available to the system.</td>
</tr>
<tr>
<td>public</td>
<td>getAvailableDevices(Device.Type type)</td>
<td>Returns an iterable collection of all devices currently available to the system by device type.</td>
</tr>
<tr>
<td>public</td>
<td>getDevice(DeviceId deviceId)</td>
<td>Returns the device with the specified identifier.</td>
</tr>
<tr>
<td>public</td>
<td>getDeviceCount()</td>
<td>Returns the number of infrastructure devices known to the system.</td>
</tr>
<tr>
<td>public</td>
<td>getDevices()</td>
<td>Returns a collection of the currently known infrastructure devices.</td>
</tr>
<tr>
<td>public</td>
<td>getDevices(Device.Type type)</td>
<td>Returns a collection of the currently known infrastructure devices by device type.</td>
</tr>
</tbody>
</table>

*YANG*

`devices` - Manage inventory of infrastructure devices

- GET /devices
- GET /devices/{id}
- GET /devices/{id}/ports

*GRPC*

# ONOSProject
Extensibility

ONOS Platform

OpenStack

Neutron

VTN Service

Endpoint Inventory

DHCP Service

#ONOSProject
### Model Annotations

Includes device friendly name, location, interface names, etc.
Model Projections

interface OpticalDevice extends Device {
    setPowerLevel();
    tuneFrequency();
}

deviceService.getDevice(id).as(OpticalDevice.class);
Multiple Layers of Abstraction

- Virtual Network
- Network Slice

Topology

- Device
- Link
- Host

Work in Progress
Network Programming

Intent
- DC Clos Fabric
- Packet/Optical WAN
- Enterprise Campus

Flow Objective
- OFDPA Pipeline
- Single Table Pipeline
- SpringOpen Pipeline

Flow Rule
- OF 1.0
- OF 1.3
- Netconf
- TL1
ONOS Intents

- ConnectivityIntent
- HostToHostIntent
- MplsIntent
- MultiPointToSinglePointIntent
- OpticalConnectivityIntent
- OpticalPathIntent
- PointToPointIntent
- SinglePointToMultiPointIntent
- TwoWayP2PIntent
- ...

#ONOSProject
**Intent Primitives**

**Primitives**
- Tunnelling / Connectivity
- Filtering
- Connectors
  - 1 to 1, 1 to M, etc.

**Domains**
- Converged packet / optical network
- DC MPLS underlay
- VXLAN overlay
- Enterprise fabric

**New Intent Service**
- Manage domain inventory
- Broker primitive requests from apps

#ONOSProject
Configuration

- **Network Configuration** (netcfg)
  - Provides mechanism for any service to register and receive configuration

- **Device Configuration**
  - **Behaviors** abstract the management and configuration aspects of a device
Northbound Roadmap

- **Intent Primitives**
  - Composable and device independent
- **Network Virtualization (hypervisor-style)**
  - Virtual Networks for Applications and/or Tenants
- **Device Configuration**
  - More generalized mechanisms for generating behaviors from models
- **Service Models**
  - Models for network services (in YANG or Tosca)

Interested in contributing? Join onos-dev@onosproject.org
ONOS Northbound API

ONOS provides a flexible and extensible API with multiple layers of abstraction for both network programming and configuration.
Software Defined Transformation of Service Provider Networks

Join the journey @ onosproject.org