



# OpenDaylight Architecture

ONF Member Work Day  
February 12<sup>th</sup>, 2015

Colin Dixon, @colin\_dixon  
TSC Chair, OpenDaylight  
Principal Engineer, Brocade

# What is OpenDaylight

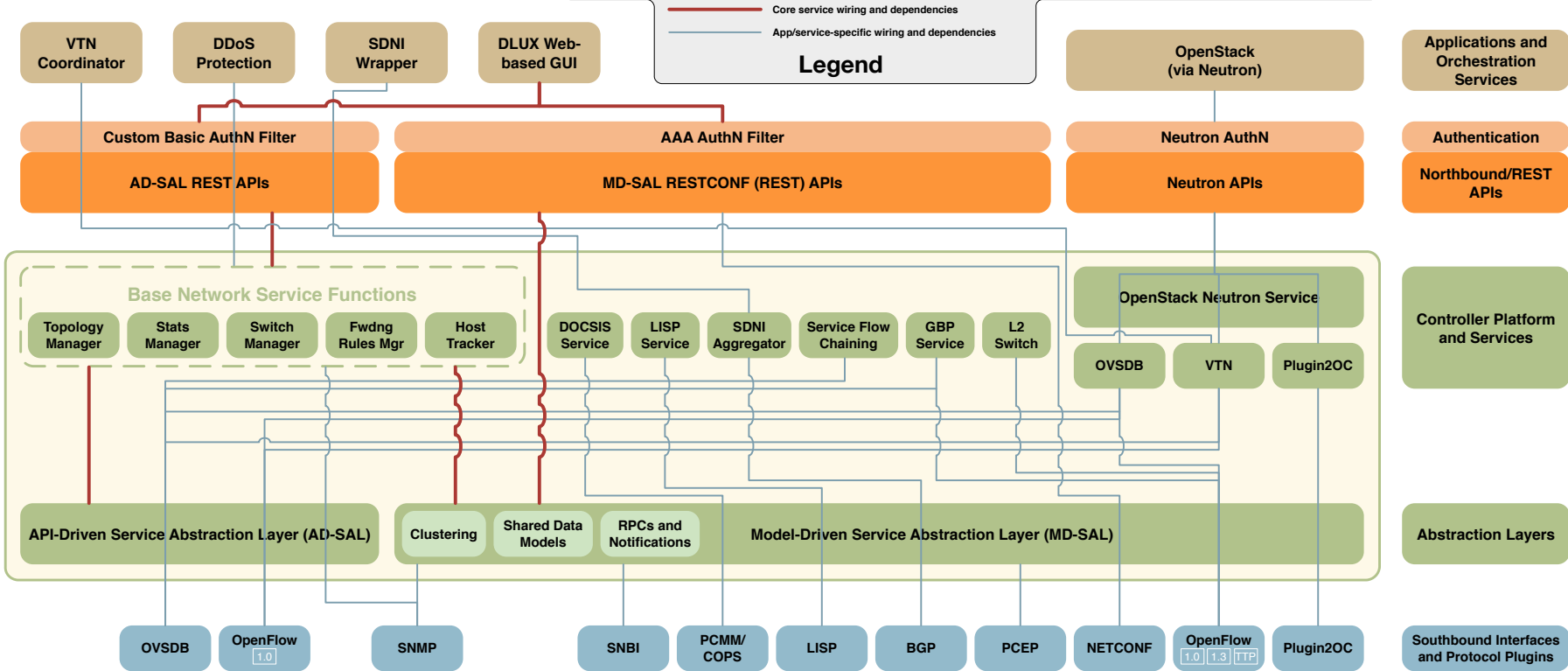
OpenDaylight is an **Open Source Software** project under the **Linux Foundation** with the goal of furthering the adoption and innovation of **Software Defined Networking (SDN)** through the creation of a common industry supported platform

Code	Acceptance	Community
To create a robust, extensible, open source code base that covers the major common components required to build an SDN solution	To get broad industry acceptance amongst vendors and users <ul style="list-style-type: none"><li>• Using OpenDaylight code directly or through vendor products</li><li>• Vendors using OpenDaylight code as part of commercial products</li></ul>	To have a thriving and growing technical community contributing to the code base, using the code in commercial products, and adding value above, below and around.



# OPEN DAYLIGHT "Helium"

**AAA:** Authentication, Authorization & Accounting  
**AuthN:** Authentication  
**BGP:** Border Gateway Protocol  
**COPS:** Common Open Policy Service  
**DLUX:** OpenDaylight User Experience  
**DDoS:** Distributed Denial Of Service  
**DOCSIS:** Data Over Cable Service Interface Specification  
**GBP:** Group Based Policy  
**LISP:** Locator/Identifier Separation Protocol  
**OVSDB:** Open vSwitch DataBase Protocol  
**PCEP:** Path Computation Element Communication Protocol  
**PCMM:** Packet Cable MultiMedia  
**Plugin2OC:** Plugin To OpenContrail  
**SDNI:** SDN Interface (Cross-Controller Federation)  
**SNBI:** Secure Network Bootstrapping Infrastructure  
**SNMP:** Simple Network Management Protocol  
**TTP:** Table Type Patterns  
**VTN:** Virtual Tenant Network



Released October, 2014  
 1.87M+ lines of code, 28 Projects, 256 Contributors

# Core Architecture

- MD-SAL
  - Model-Driven Service Abstraction Layer
- Components interact via YANG models
  - RPCs
  - Notifications
  - Data Store

# Core Architecture

- Models are layered to increase abstraction
  - Path programming model
    - (over)
  - Flow programming model
    - (over)
  - OpenFlow model
- Common models provide unifying information to write many apps
  - Topology, Inventory, Neutron, etc.

# Example YANG Model

```
container network-topology {
  description "...";
  key "topology-id";
  leaf topology-id {
    type topology-id;
    description "...";
  }

  list node {
    description "...";
    key "node-id";
    uses node-attributes;
  }

  list link {
    description "...";
    key "link-id";
    uses link-attributes;
  }
}
```

- Network Topology
- List of Nodes
- List of Links
- Links and Nodes can be “extended” later
- Can specify constraints

# YANG Models as an Architecture

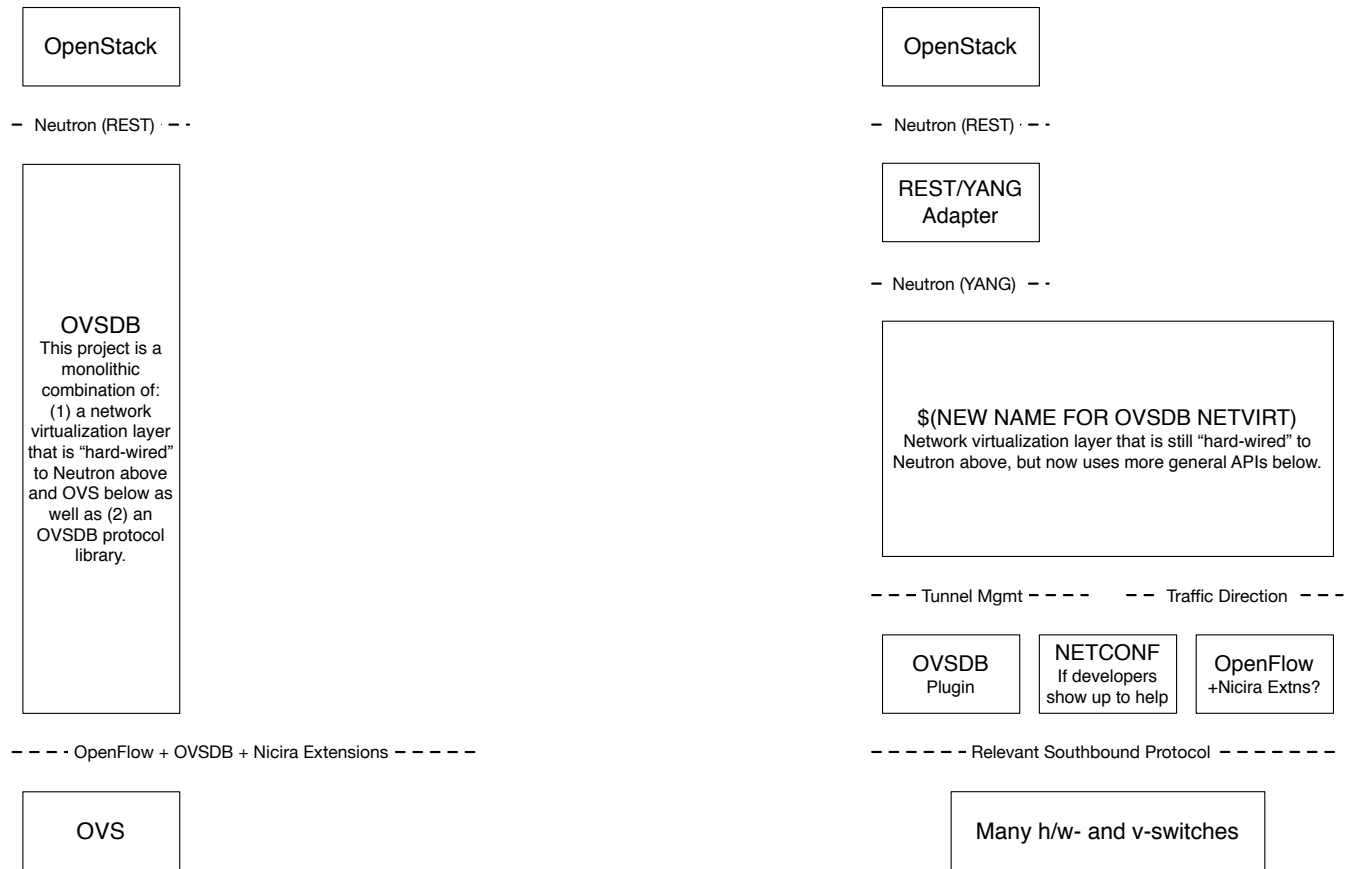
## ▪ Good

- Easy to augment common models
  - Some apps just work
- Modeling is great
  - Easily defined APIs between projects
  - You can do real work
- Native NETCONF
- Good Java tooling

## ▪ Harder

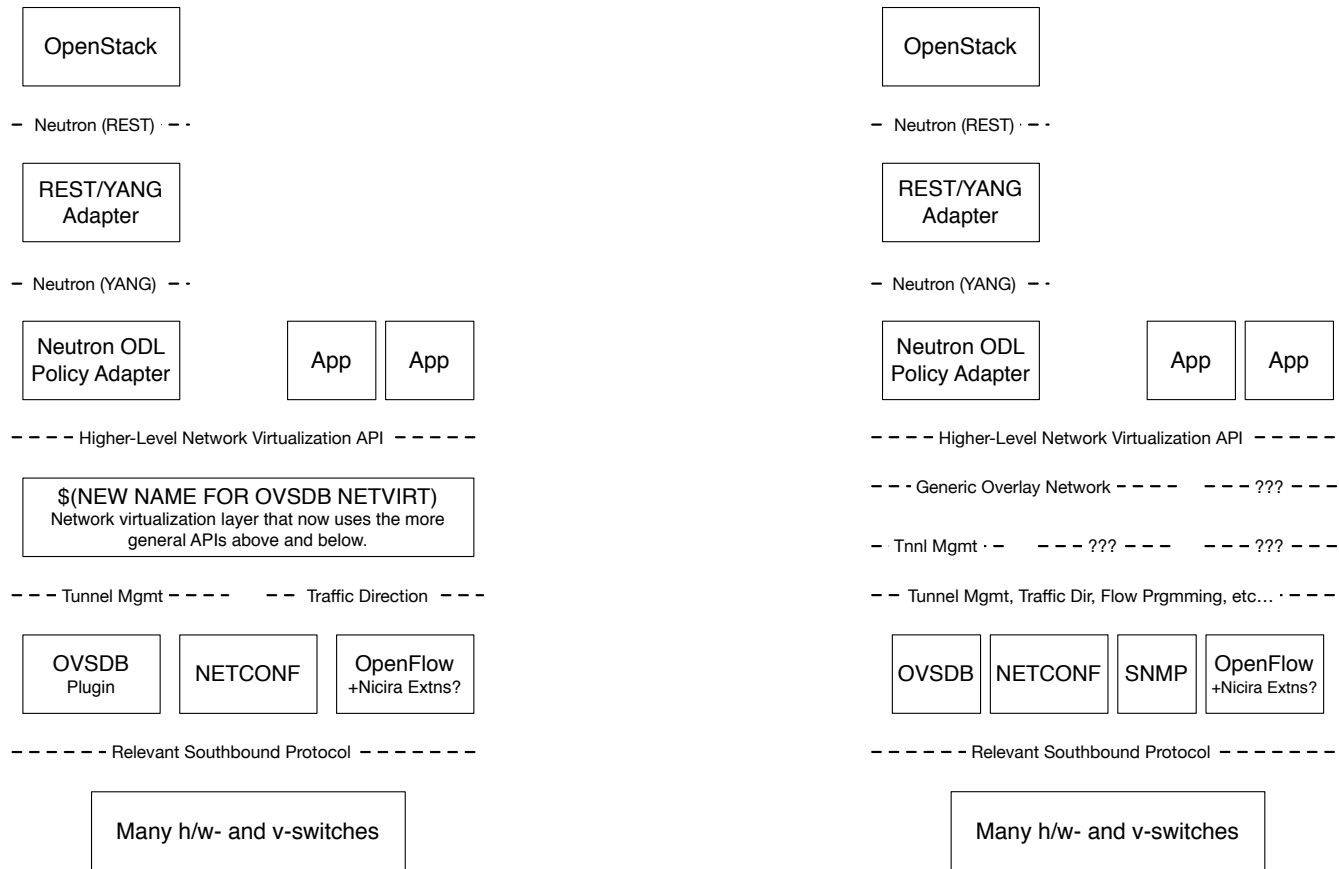
- Interaction via shared *data* models is hard
  - No status codes
- RPCs fix that, but can be less flexible
- YANG isn't perfect
  - No recursive self-inclusion
  - Less tools than alternatives

# Architecture in Evolution (OVSDB/Neutron)





# Architecture in Evolution (OVSDB/Neutron)





# Get Involved

A community-led and industry-supported open source platform to advance Software-Defined Networking (SDN) and Network Functions Virtualization (NFV).

- Pull the code and review documentation at [wiki.opendaylight.org](http://wiki.opendaylight.org)
- Connect with active developers in the community on the #opendaylight IRC channel at freenode.net [webchat.freenode.net](http://webchat.freenode.net)
- Join the conversation through [lists.opendaylight.org](http://lists.opendaylight.org) and [ask.opendaylight.org](http://ask.opendaylight.org)
- Propose a new project at [wiki.opendaylight.org/view/Project\\_Proposals:Main](http://wiki.opendaylight.org/view/Project_Proposals:Main)