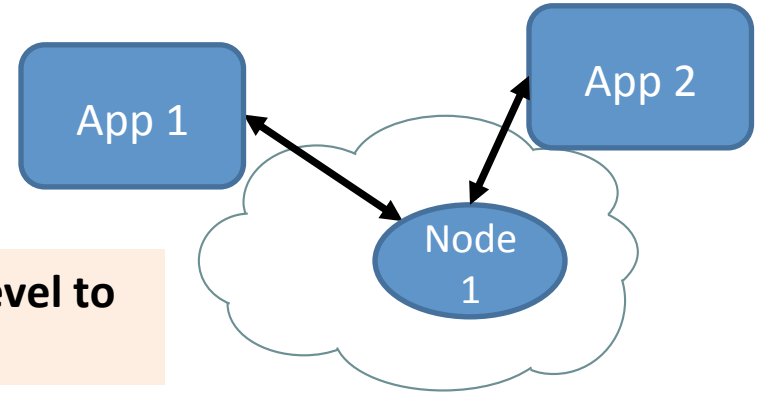
Why Intent-Driven Network Interfaces?

Application needs Intent-Driven not prescriptive Control

- Application to state:
  - A connection between two sites with flows
  - A service flow with SLA
  - A customer network service chain
  - **Security level for data**
- Intent Driven: What I want not how to do it
  - Let network layers figure out how to accomplish intent
- High level
  - Yang is low-level specific to device

Applications need a Simple API

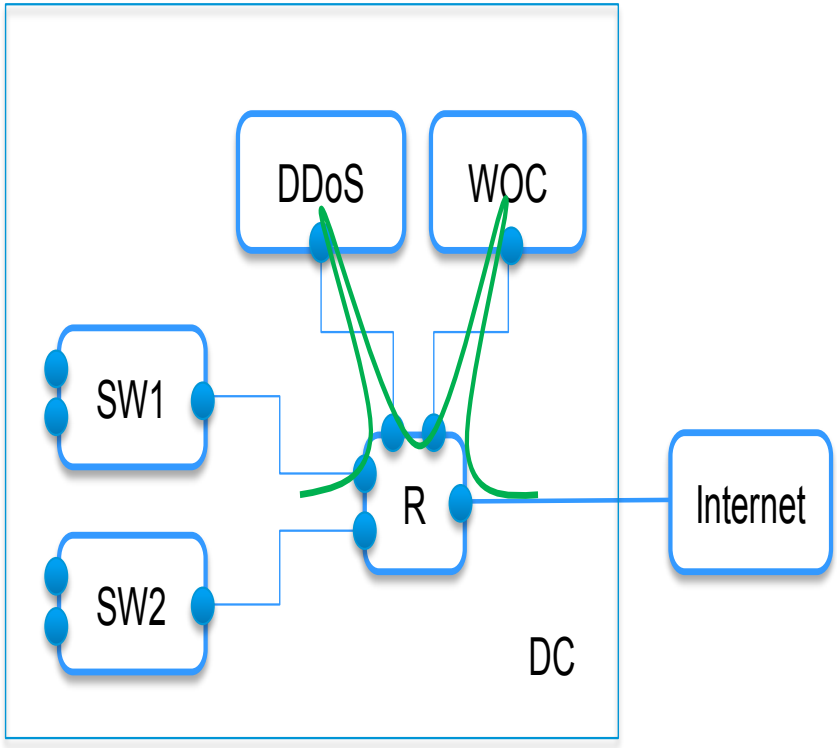
- Request virtual networks through specific nodes with network services at flow rate,
- When applications can aid control of network, storage, compute – can reach 95% utilization of net, storage, compute
- **NeMo for Virtual network** 3 primitive groups, 15 sentences, and 36 key words



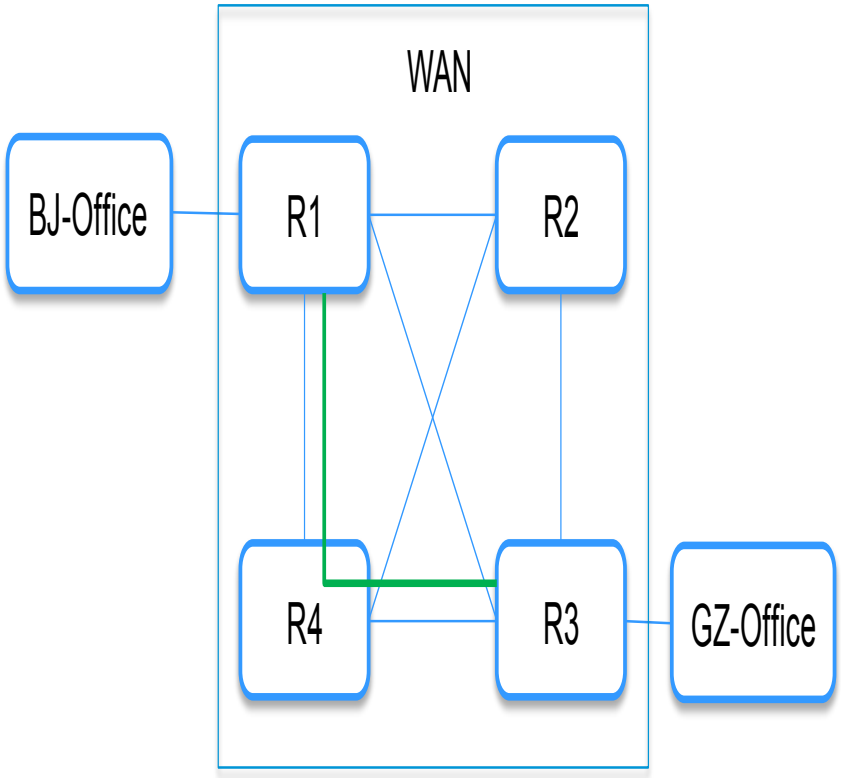
NeMo wants to security level to Virtual network

2 sample Interfaces

Service Chaining



Virtual WAN with TE

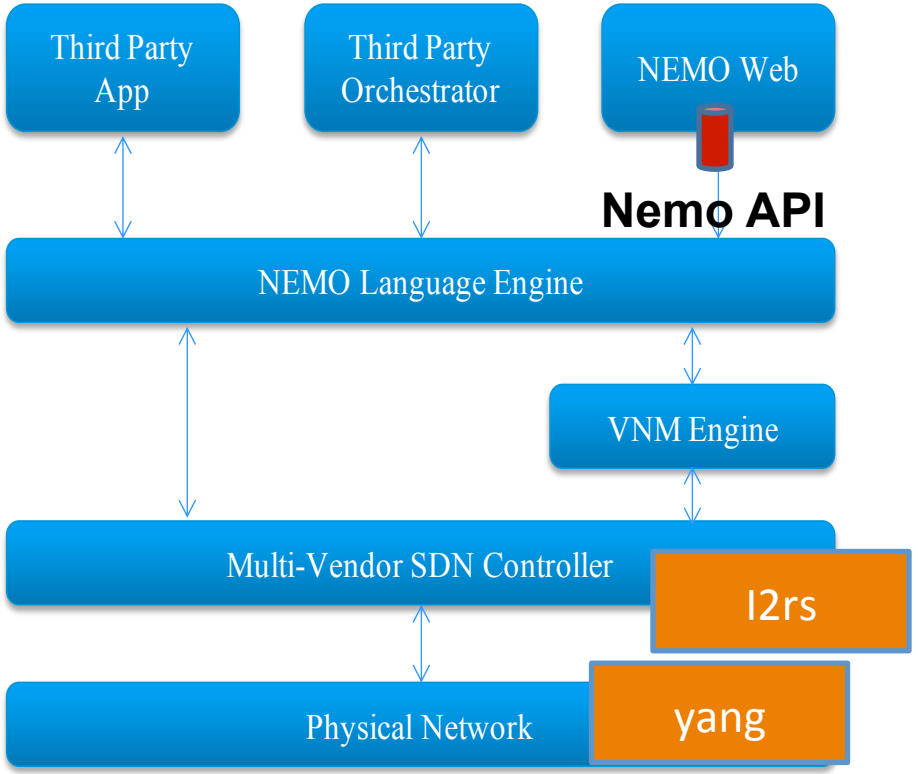


NeMo allows Multi-Service SDN

MultiService SDN controller

- **Problem**
  - It's hard to support multiple, independently developed SDN applications or services without **resource conflicts**
- **State of the Art**
  - ODL Helium has not solved this problem which prevents competing flow writers that can't be run simultaneously.
  - It is not possible to run e.g. NetVirt and SFC services in the same controller domain.
  - Commercial controllers have not solved this problem either

NeMo's API uses REST/RPC to talk to Nemo Language Engine



NEMO Language: Concise and Flexible

Resource Access

Entity Model	**node**	Node/UnNode entity_id Type {FN|PN|LN} Owner node_id Properties key1 ,value1
	link	Link/UnLink entity_id Endnodes (node1_id,node2_id) SLA key,value Properties key1 ,value1 ….
	flow	Flow/UnFlow entity_id Match/UnMatch key1, value1|Range(value, value) | Mask(value, value) Properties key1 ,value1

Policy and Event Handling

Capability Model	**Query**	Query key Value {value} From entity_id
	Policy	Policy/UnPolicy policy_id Appliesto entity_id Condition {expression} Action { "forwardto" | "drop" | "gothrough" | "bypass" | "guaranteeSLA" | "Set" |"Packetout"| Node | UnNode | Link | Unlink} Commit / Withdraw
	Notifica-tion	Notification entity_id On key Every period RegisterListener callbackfunc

Model Definition and Transactions Control

Node definition	**NodeModel <node_type> Property { <data_type> : <property_name> }**
Link definition	**LinkModel <Link_type> Property { <data_type> : <property_name> }**
Action definition	**ActionModel <Action_Name> parameter { <data_type> : <property_name> }**
Connect	**Connect <conn-id> Address <ip-prefix> Port <integer>**
	Disconnect <conn_id>
Transaction	**Transaction …. Commit**