Modeling Language Analysis for Network Virtualization: YANG vs. TOSCA  
  
by  
Himamshu Aluru  
20159595

**Introduction**

As network infrastructures become increasingly virtualized and software-defined, the importance of standardized, extensible, and machine-readable data modeling languages becomes paramount. These languages facilitate automation, configuration, orchestration, and validation of network services and devices. Among the leading modeling languages are **YANG (Yet Another Next Generation)** and **TOSCA (Topology and Orchestration Specification for Cloud Applications)**. This document presents a comparative analysis of these two modeling languages with respect to their applicability in network virtualization scenarios.

**Key Syntax Overview**

**YANG Syntax (RFC 6020 / RFC 7950)**

YANG is a data modeling language used to model configuration and state data manipulated by the Network Configuration Protocol (NETCONF). It is tree-structured and follows a modular syntax.

**Example: Basic Interface Configuration in YANG**

A screen shot of a computer screen

AI-generated content may be incorrect.  
  
**TOSCA Syntax (YAML-based)**

TOSCA describes service templates using a YAML-based syntax to define cloud applications' topology, components, relationships, and orchestration policies.

**Example: Simple Compute Node in TOSCA**

A computer screen shot of a black screen

AI-generated content may be incorrect.

**Detailed Instructions and Step-by-Step Analysis**

**Step 1: Select Modeling Languages**

* **Selected**: YANG and TOSCA
* **Reason**: Widely adopted in network configuration (YANG) and orchestration (TOSCA)

**Step 2: Define Use Case - Network Virtualization**

We assume the use case of virtualizing a traditional network setup with multiple routers, switches, and middleboxes managed through SDN/NFV principles.

**Step 3: Evaluate Criteria**

|  |  |  |
| --- | --- | --- |
| **Criterion** | **YANG** | **TOSCA** |
| **Primary Focus** | Configuration management | Application/service orchestration |
| **Data Format** | XML / JSON | YAML |
| **Adopted By** | IETF, NETCONF, OpenConfig | OASIS, OpenStack, Cloudify |
| **Tooling Support** | pyang, yanglint, sysrepo | TOSCA parser, Alien4Cloud |
| **Extensibility** | High (module/submodule) | High (custom node types) |
| **Integration with SDN/NFV** | Strong (OpenDaylight, ONOS) | Moderate (via orchestrators) |
| **Learning Curve** | Moderate to steep | Moderate |

**Step 4: Apply to Realistic Example**

**YANG Use Case: Automating Router Configuration**

A YANG model can be used with NETCONF to automate the configuration of multiple routers in a virtualized lab.

**Sample Scenario:**

A black background with blue and white letters

AI-generated content may be incorrect.  
Where config.xml is generated from YANG schema.

**TOSCA Use Case: Deploying NFV Service Chain**

A TOSCA template defines a virtual firewall and IDS and their relationships in a service graph.

**Sample Scenario:**



**Step 5: Perform Comparative Evaluation**

* **YANG** excels in low-level configuration tasks and is ideal for modeling device data for network elements.
* **TOSCA** excels in defining cloud service topologies and is more suitable when multiple components must be orchestrated together.

**Security Considerations**

* **YANG** supports strong model validation via constraints (must, when, range) and enforces type safety.
* **TOSCA** allows policy-driven enforcement (e.g., access, scaling) but depends heavily on the orchestrator for secure execution.

**Visual Aids**

**SDN/NFV Stack Integration Diagram**

A computer screen shot of a program

AI-generated content may be incorrect.  
 **Feature Comparison Table**

See earlier table under "Evaluate Criteria"

**Appendix**

**Full YANG Configuration Snippet**

A screen shot of a computer

AI-generated content may be incorrect.  
 **Full TOSCA YAML Snippet**

A screen shot of a computer

AI-generated content may be incorrect.

**Critical Thinking / Originality**

**Hybrid Use Case Proposal**

An enterprise could use TOSCA to define and orchestrate a full SDN/NFV service (e.g., load balancer + IDS + analytics), while using YANG to configure each virtual network function (VNF) individually. This separation of concerns ensures scalability and granular control.

**Evaluation Matrix**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **YANG** | **TOSCA** |
| Tool Support | 4.5 | 3.5 |
| Integration Readiness | 5.0 | 4.0 |
| Learning Curve | 3.0 | 4.0 |
| Extensibility | 5.0 | 4.5 |
| Industry Adoption | 4.5 | 3.5 |
| **Total (out of 25)** | **22.0** | **19.5** |

**Conclusion**

Both YANG and TOSCA play critical roles in enabling network virtualization, but they serve different layers of the stack. YANG is ideal for configuring and managing network devices and services via protocols like NETCONF and RESTCONF. In contrast, TOSCA is suited for modeling and orchestrating network services and applications across virtualized infrastructures.

**Recommendation**: For an organization focusing on **device-level network virtualization**, **YANG** is the preferred modeling language due to its maturity in network configuration. However, if **service-level orchestration** across hybrid environments is a goal, **TOSCA** should be integrated via orchestrators like Cloudify or ONAP.

**References**

1. Bjorklund, M. (2010). RFC 6020: YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF). IETF.
2. Bjorklund, M. (2016). RFC 7950: The YANG 1.1 Data Modeling Language. IETF.
3. OASIS. (2020). TOSCA Simple Profile in YAML Version 1.3.
4. OpenDaylight Project: <https://www.opendaylight.org/>
5. Cloudify Documentation: <https://docs.cloudify.co/>
6. pyang GitHub Repository: <https://github.com/mbj4668/pyang>
7. ONAP Project: <https://www.onap.org/>