

# Overview


- About Model-Driven Programmability, on page 1
- About the Programmable Interface Infrastructure, on page 1


# About Model-Driven Programmability

The model-driven programmability (MDP) of Cisco NX-OS device allows you to automate the configuration and control of the device. It also allows you to monitor the state changes of the device.

Data modeling provides a programmatic and standards-based method of writing configurations to the network device, replacing the process of manual configuration. Data models are written in a standard, industry-defined language. Although configuration using a CLI may be more human-friendly, automating the configuration using data models results in better scalability.

The Cisco NX-OS device supports the YANG data modeling language. YANG is a data modeling language used to describe configuration and operational data, remote procedure calls, and notifications for network devices.

The following are standards-based programmable interfaces supported by NX-OS:

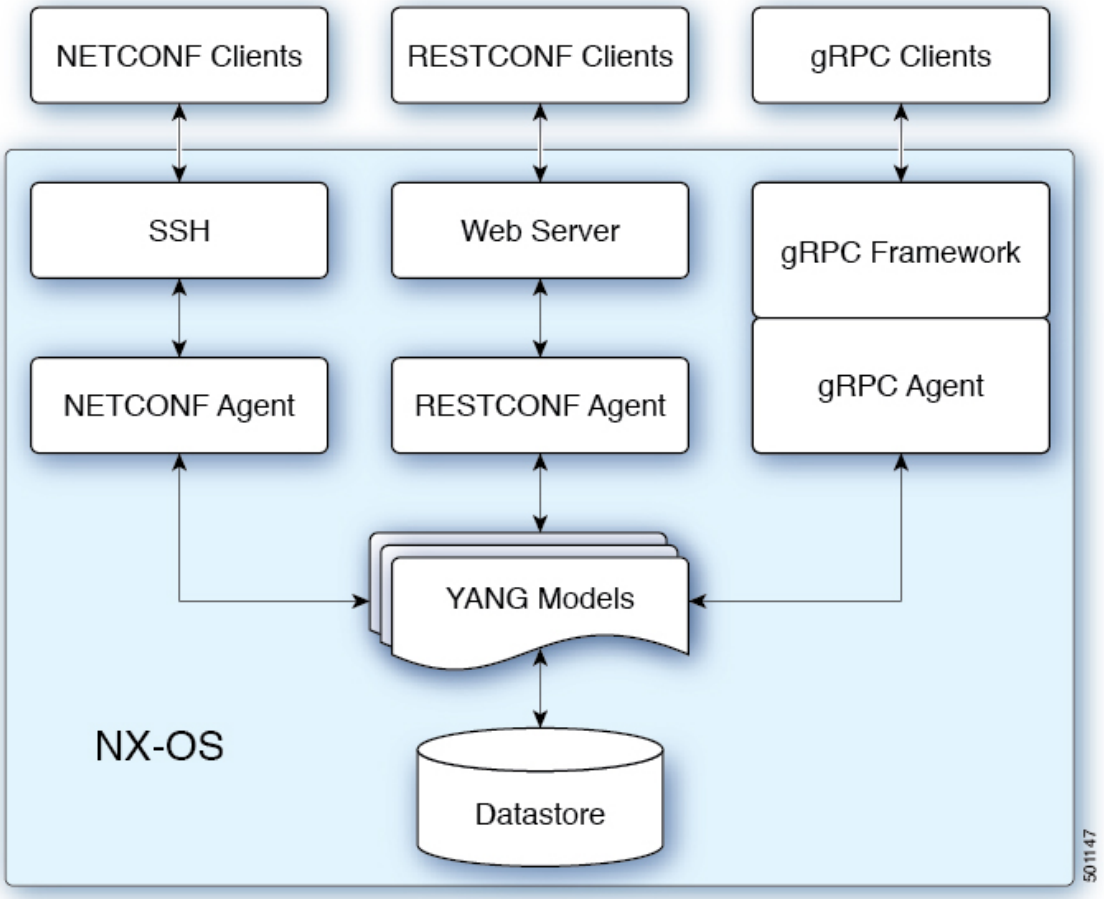
- NETCONF
- RESTCONF
- gRPC

The following are observability interfaces supported by NX-OS:

- NETCONF
- gRPC
- Telemetry

# About the Programmable Interface Infrastructure

This section provides a brief overview of the NX-OS Programmable Interface infrastructure.




When a request is received through NETCONF, RESTCONF, or gRPC, the request is converted into an abstract message object. That message object is distributed to the underlying model infrastructure based on the model namespace in the request. Using the namespace, the appropriate model is selected, and the request is passed to it for processing. The model infrastructure executes the request (read or write) on the underlying NX-OS device datastore. The results are returned to the agent of origin for response transmission back to the requesting client.

# NX-OS Programmability Agents

Agents provide an interface between the NX-OS device and the external clients. They specify the specific transport, the protocol, and the encoding of the communications with the networking device. NX-OS Programmable Interfaces support three agents: NETCONF, RESTCONF, and gRPC, each providing different interfaces for configuration management via YANG models.

***Table 1: NX-OS Programmable Interface Agents***

| Agent | Transport | Protocol | Encoding |
|---|---|---|---|
| NETCONF | SSH | NA | XML |
| RESTCONF | HTTP | draft-ietf-netconf-restconf-10[1] | XML or JSON |

| Agent | Transport | Protocol | Encoding |
|---|---|---|---|
| gRPC | HTTP | gRPC Protocol [2] | Google Protobuf |

The protocol specifications are described in the following documents:

- RESTCONF Agent
- gRPC Agent

# NX-OS Observability Agents

Note that there exist read-only streaming interfaces which serve to monitor the switch events. NX-OS supports standard-based NETCONF, gRPC, and the proprietary software telemetry.

***Table 2: NX-OS Observability Interface Agents***

| Agent | Transport | Protocol |
|---|---|---|
| NETCONF | SSH | NA |
| gRPC | HTTP | gRPC Protocol |
| Telemetry | HTTP, UDP, GRPC | NA |

# Model Infrastructure

The Model Infrastructure takes requests that are received from the Agent, determines the namespace that is associated with the YANG model in the request, and selects the model component matching the namespace to process the request. When the selected model component completes request processing, the processing results are sent to the requesting Agent for transmission back to the client. The Model Infrastructure is also responsible for handling protocol initiation requests involving authentication, handshaking, and so on, as specified by the Agent protocol.

# Native (Device) YANG Model

The YANG model, where the Device Configuration is described, is called a device YANG model. The Device Model is manifested in the Model Infrastructure as another model component with the device namespace.

# Common (OpenConfig) YANG Models

A Common Model is another kind of model component defined by the public community. The YANG paths in the common model could translate to the equivalent Device Model elements. These equivalent Device Model elements are used to read and write Device Model data in the Device YANG context.

**Note** Cisco NX-OS maintains a centralized DME database. The Device and OpenConfig YANG are the wrapping representation of the DME database. This guarantees consistency across all programmability interfaces. Users of different interfaces would see the same config/state of the switch without discrepancy

# Additional YANG References

For additional information about YANG, see https://handwiki.org/wiki/YANG.