



NESi

Network Equipment Simulator

The background features a complex, glowing blue network of nodes and connections. The nodes are represented by small, bright blue dots, and they are interconnected by thin, glowing blue lines. The overall effect is a dense, interconnected web of light against a dark, almost black background. The text "Our Goals" is centered in the middle of this network.

Our Goals

Our Goals

- Simulation of certain points of a network (DSLAMs as of now)
 - Simulating the interaction between cards, ports and CPEs
- Offering vendor specific management interfaces supporting command line operations
- Creating a testing environment for network management and automation

The image features a complex, glowing blue network of nodes and connections against a dark background. The nodes are represented by small, bright blue dots, and they are interconnected by thin, glowing blue lines. The network is dense and irregular, with many nodes having multiple connections. The overall appearance is that of a digital or data network, possibly representing a social network, a data structure, or a complex system. The word "History" is centered in the image in a white, sans-serif font.

History

From Softboxen...

- NESi originates from Ilya Etingof's Softboxen (<https://github.com/etingof/softboxen>)
- Started as a basic CLI simulation of a dummy Cisco Switch
- Just the basic CLI mode
- Basic CLI functionalities
- Heavily relied on tox

...to NESi

NESi now supports:

- lightweight debugging options
- a lightweight testing framework
- multiple network socket modes
- Advanced CLI options (history, arrow navigation, etc.)

The image features a complex, glowing blue network of nodes and connections against a dark background. The nodes are represented by small, bright blue spheres, and they are interconnected by thin, glowing blue lines. The network is dense and multi-layered, with many nodes having multiple connections, creating a web-like structure. The overall appearance is that of a digital or neural network. The word "Architecture" is centered in the image in a white, sans-serif font.

Architecture

REST API

- Software is backed by a REST API
- API has to run permanently in the background (as of now)
- API is the communication backbone between database and CLI
- Allowing the user to create, clone or delete devices and sub-components
- API structure is backed by the flask python module

REST API - Overview

```
@app.route(PREFIX + '/boxen/<box_id>/ports', methods=['GET'])
def show_ports(box_id):
    if flask.request.args is None:
        req = {}
    else:
        req = flask.request.args

    response = show_components(PortsSchema(), Port, req, box_id)
    return response, 200

@app.route(PREFIX + '/boxen/<box_id>/ports/<id>', methods=['GET'])
def show_port(box_id, id):
    response = show_component(Port, box_id, id)
    return response, 200

@app.route(PREFIX + '/boxen/<box_id>/ports', methods=['POST'])
def new_port(box_id):
    req = flask.request.json
    ...
```

REST API Structure:

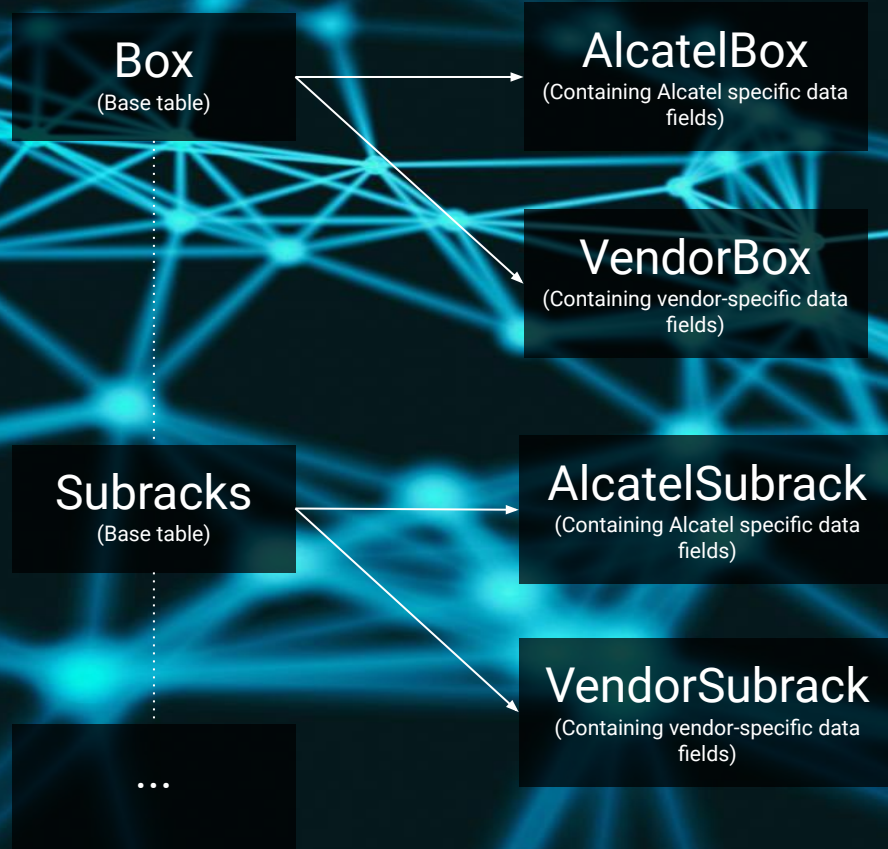
<https://127.0.0.1:5000/nesi/v1/boxen/<device-id>/...>

- ...subracks/<subrack-id>
- ...cards/<card-id>
- ...ports/<port-id>
- ...

Database

- SQLite as lightweight database solution in combination with python's marshmallow library
- Dynamically created on each API launch
- Database structures backed by models and schema files
- Database is populated via shell scripts
- Base tables extended by vendor specific tables

Database - Overview



CLI

- CLI backend written in Python
- CLI can be operated in either daemon or gui mode
- Retrieving data from DB and dynamically filling Jinja2 templates to generate output
- CLI is based on a REPR loop
- Advanced CLI operations (local history, arrow navigation)

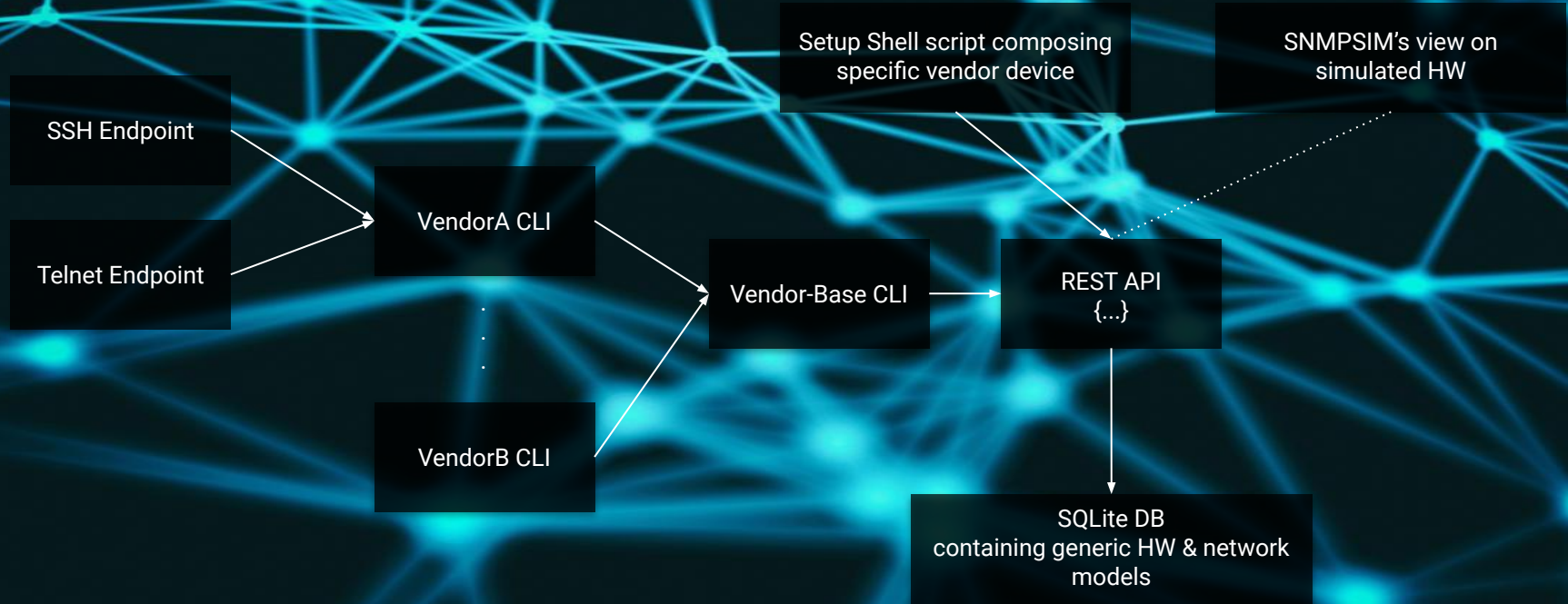
REPR Loop

Read - Evaluate - PRint

The command processor implementing this loop operates like the following:

- Reading user input up to <CR>
- Try matching input against known CLI commands
- Applying command to underlying device
- Locating correct Jinja2 template and presenting it to the user

Overview





Vendors

Alcatel



ALCATEL

login:admin

password:

Welcome to Alcatel_7360

last login : 01/03/2021 14:09:57

Alcatel_7360>#

Huawei



```
>>User name:root  
>>User password:
```

```
Huawei Integrated Access Software (MA5623).  
Copyright(C) Huawei Technologies Co., Ltd. 2002-2015. All rights reserved.
```

```
-----  
User last login information:  
-----
```

```
Access Type : telnet  
IP-Address  : 10.0.0.12  
Login Time  : /  
Logout Time : /  
-----
```

```
Huawei_5623A>█
```


KeyMile



```
login as: manager
```

```
password: *****
```

```
---===### CLI Release R2A20, Build 2014-01-31 ###===---
```

```
/> █
```

EdgeCore

```

● ● ●

>>User name:admin
>>User password:
*****
WARNING - MONITORED ACTIONS AND ACCESSES
FNT

Station's information:

Floor / Row / Rack / Sub-Rack
 / / /
DC power supply:
Power Source A: Floor / Row / Rack / Electrical circuit
 / / /

Number of LP:
Position MUX:
IP LAN: 127.0.0.1
Note: ed-ge-co-re-1
MOTD:
*****

      CLI session with the ECS4120-28Fv2-I is opened.
      To end the CLI session, enter [Exit].
ed-ge-co-re-1>|
```

The image features a complex, glowing blue network of nodes and connections against a dark background. The nodes are represented by small, bright blue spheres, and they are interconnected by thin, glowing blue lines. The network is dense and irregular, with many nodes having multiple connections. The overall effect is that of a digital or neural network. The word "Demo" is centered in the image in a white, sans-serif font.

Demo





The Future of NESi

Features to come

- Separated databases for each vendor
- Separation of API and database
- Improved exception-handling
- Python auto-docs
- Improved telnet socket
- Separated physical and logical layer
- Implementation of SNMPSIM

SNMPSIM

- Can simulate many thousands of different SNMP speaking devices
- Runs over IPv4 and/or IPv6 transports
- Varies response based on SNMP Community, Context, source/destination addresses and ports
- Offers REST API based control plane

in combination with NESi

- Possibility to simulate states of NESi devices from the SNMP point of view
- Implementation as either python module or built in stripped down backend

The background features a complex, glowing blue network of nodes and connections. The nodes are represented by small, bright blue dots, and they are interconnected by thin, glowing blue lines. The overall effect is a dense, interconnected web of light, suggesting a network or a complex system. The text "Question & Feedback" is centered in the middle of the image in a white, sans-serif font.

Question & Feedback

A glowing blue network of nodes and connections on a dark background. The nodes are represented by small, bright blue dots, and they are interconnected by thin, glowing blue lines, creating a complex, web-like structure. The overall effect is that of a digital or neural network, with a sense of depth and connectivity.

Looking forward to your contributions!

NESi on GitHub

<https://github.com/inexio/NESi>



NESi-UI on GitHub

<https://github.com/inexio/NESi-UI>



SNMPSIM on GitHub

<https://github.com/inexio/snmpsim>



The background features a complex, glowing blue network of nodes and connections. The nodes are represented by small, bright blue dots, and they are interconnected by thin, glowing blue lines. The overall effect is a dense, interconnected web of light against a dark, almost black background. The lines and nodes vary in brightness, creating a sense of depth and activity.

Thank you!