NETWORK AUTOMATION FORUM

JUNIPER NETWORKS

Automated Doc Generation

Jose Miguel Izquierdo

Technical Marketing Engineer @ Juniper Networks

AUTOCON 2

THE NETWORK AUTOMATION CONFERENCE

Jose Miguel Izquierdo

- 12 years at Juniper
    - 10y as Professional Services Consultant
    - 2y as Technical Marketing Engineer
- Based in Madrid, Spain (EMEA)
- linkedin.com/in/josemiguelizquierdo

JoseMi

- 12 years at Juniper
    - 10y as Professional Services Consultant
    - 2y as Technical Marketing Engineer
- Based in Madrid, Spain (EMEA)
- linkedin.com/in/josemiguelizquierdo

Agenda

- Challenges
- Motivations
- Solution
- Tools & Examples
- Use cases (2x)

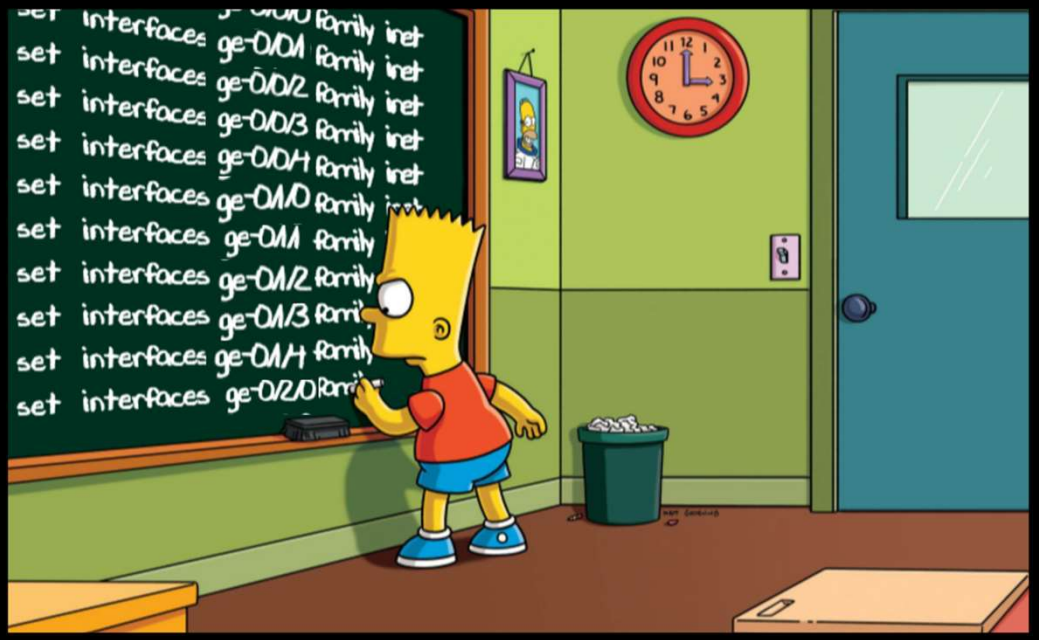
Manual Documentation challenges

- Time-consuming

- Prone to human error

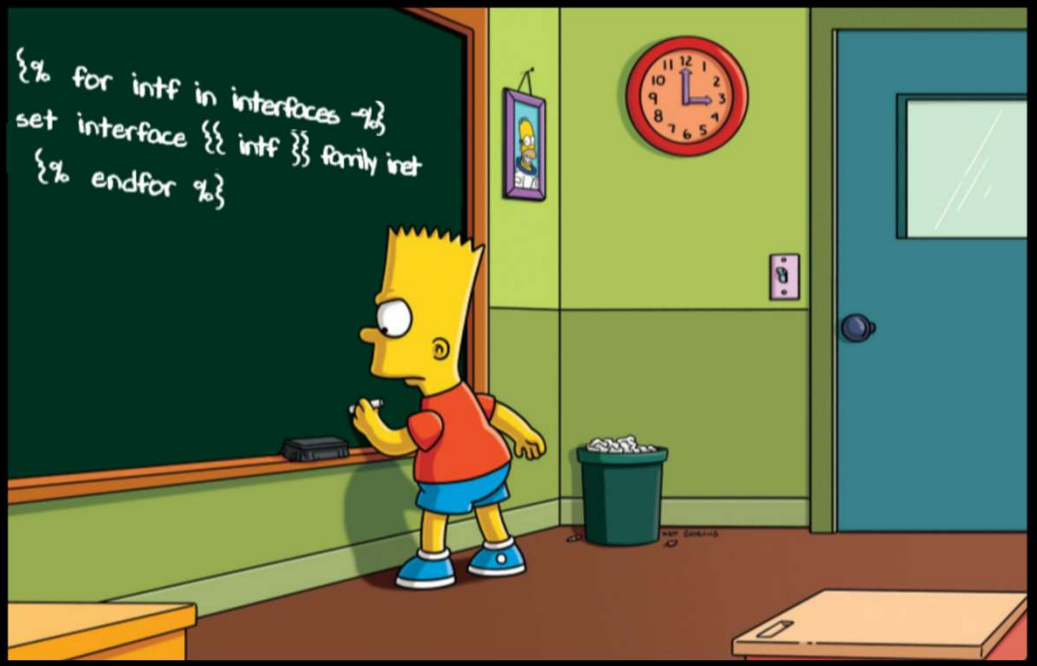
- Inconsistent across iterations and projects

- Difficulty in maintaining version control



Motivations

- Standardization
- Consistent look & feel
- Tracking control
- Change management
- Reusability
- Integration



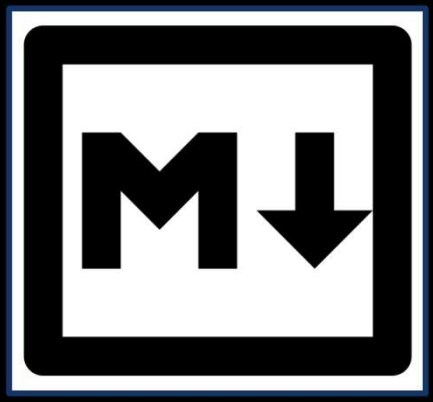
Solution

Combining tools

- Markdown
- Mermaid
- PlantUML
- Jinja2
- Ansible
- Docker
- Jenkins
- Git
- Version Control systems
- Pandoc
- LaTeX
- AI

Markdown

- Lightweight markup language with plain text formatting syntax.
- Easy to write & read (just a few special chars)
- Simplicity
- Portability (easily converted to pdf, word, etc…)
- Works with almost any text editor



Markdown examples I

```
1
2  # Heading 1
3  ## Heading 2
4  ### Heading 3
5
6  [www.google.es](https://www.google.es) - clickable links (URL)
7
8  [x.cfg](./files/x.cfg) - clickable links (open local files)
9
10 ![image](./images/JNPR White Background.png) - images
11
12 ## Lists
13
14 1. Ordered list
15 2.
16
17 - Unordered list
18 -
```

Heading 1

Heading 2

Heading 3

www.google.es - clickable links (URL)

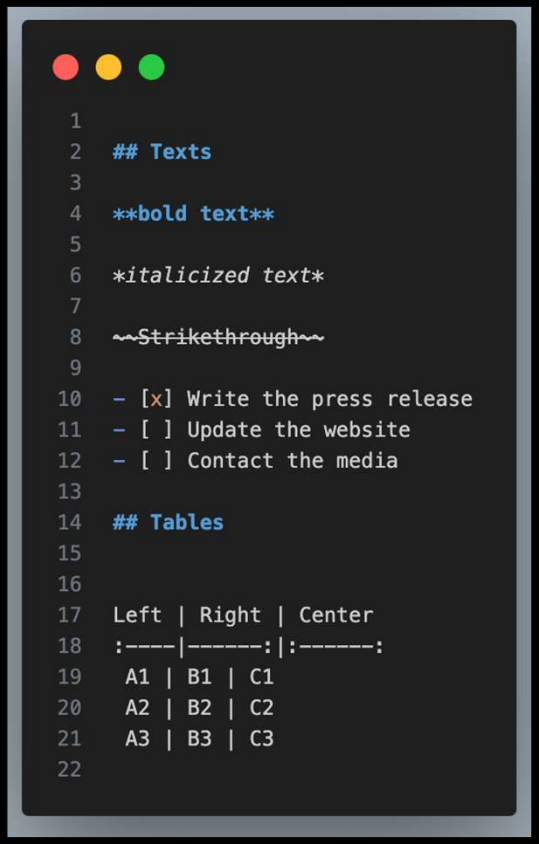
x.cfg - clickable links (open local files)

Lists

1. Ordered list
2.

- Unordered list
-

Markdown examples II



```
## Texts

**bold text**

*italicized text*

~~Strikethrough~~

- [x] Write the press release
- [ ] Update the website
- [ ] Contact the media

## Tables


Left | Right | Center
:----|------:|:------:
 A1 | B1 | C1
 A2 | B2 | C2
 A3 | B3 | C3

```

Texts

bold text

italicized text

~~Strikethrough~~

- [x] Write the press release
- [ ] Update the website
- [ ] Contact the media

Tables

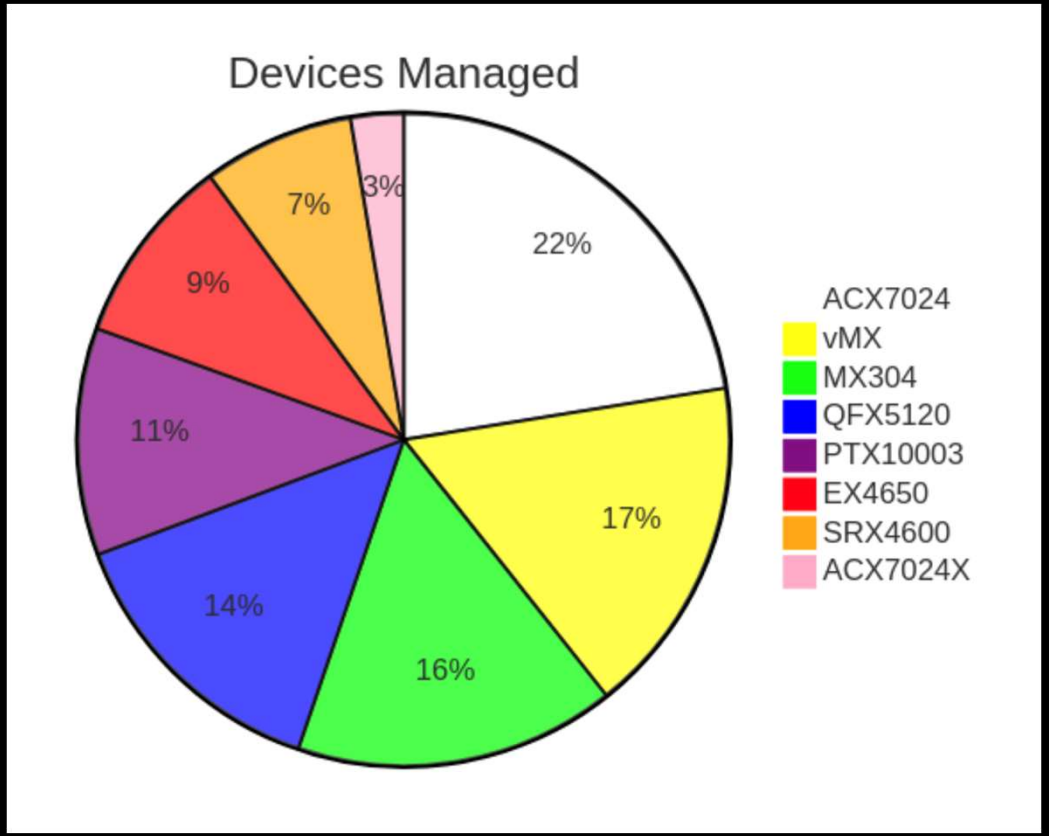
Left	Right	Center
A1	B1	C1
A2	B2	C2
A3	B3	C3

Mermaid

- Diagrams As Code
- Simple
- Dynamic
- Easy to learn, adopt…
- Version control friendly
- Live editors
- https://mermaid.js.org/

Mermaid example

```mermaid
%%{init: {'theme': 'base',
'themeVariables': { 'pie1': '#FFFFFF',
                    'pie2': '#FFFF00',
                    'pie3': '#00FF00',
                    'pie4': '#0000FF',
                    'pie5': '#800080',
                    'pie6': '#ff0000',
                    'pie7': '#FFA500'}}}%%
pie title Devices Managed
  "ACX7024" : 120
  "ACX7024X" : 14
  "MX304" : 85
  "QFX5120" : 75
  "EX4650" : 50
  "SRX4600" : 40
  "PTX10003" : 60
  "vMX" : 90
```



PlantUML

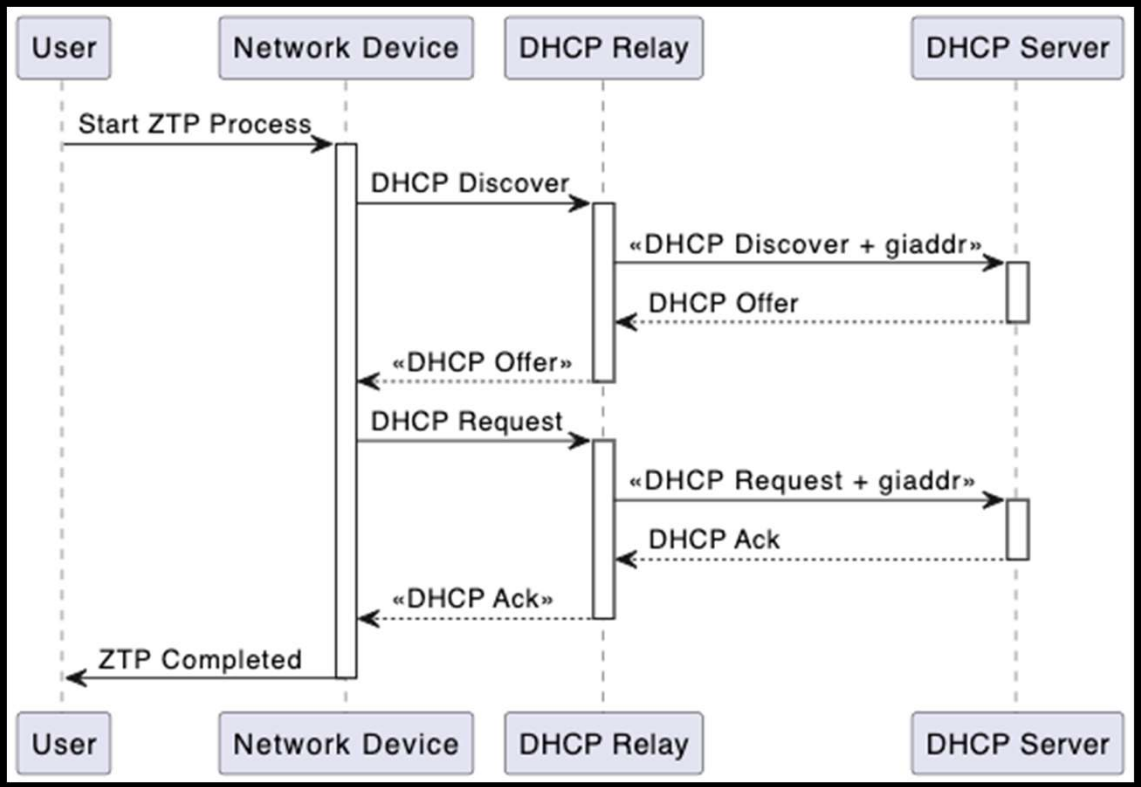
- Diagrams As Code
- Simple
- Dynamic
- Easy to learn, adopt…
- Version control friendly
- Live editors
- https://plantuml.com/

PlantUML example

```plantuml
@startuml
participant User
participant "Network Device" as ND
participant "DHCP Relay" as PS
participant "DHCP Server" as CM

User -> ND: Start ZTP Process
activate ND

ND -> PS: DHCP Discover
activate PS
PS -> CM: << DHCP Discover + giaddr>>
activate CM
CM --> PS: DHCP Offer
deactivate CM
PS --> ND: << DHCP Offer >>
deactivate PS
ND -> PS: DHCP Request
activate PS
PS -> CM: << DHCP Request + giaddr>>
activate CM
CM --> PS: DHCP Ack
deactivate CM
PS --> ND: << DHCP Ack >>
deactivate PS
ND -> User: ZTP Completed
deactivate ND
@enduml
```



Jinja2

- Templating engine
- Templates rendering:
    - variables {{ }}
    - logic {% … %} (Loops and conditionals)
- Any text artifacts
- Efficiency
- Reusability

Jinja2 examples I

```
{% for interface in interfaces %}
set interfaces {{ interface.name }} description "{{ interface.description }}"
set interfaces {{ interface.name }} unit {{ interface.unit }} vlan-id {{ interface.vlan_id }}
set interfaces {{ interface.name }} unit {{ interface.unit }} family inet address {{ interface.ipv4_address }}
{% endfor %}
```

```
set interfaces ge-0/0/0 description "Uplink to Core"
set interfaces ge-0/0/0 unit 0 vlan-id 100
set interfaces ge-0/0/0 unit 0 family inet address 192.168.1.1/24

set interfaces ge-0/0/1 description "LAN interface"
set interfaces ge-0/0/1 unit 0 vlan-id 200
set interfaces ge-0/0/1 unit 0 family inet address 192.168.2.1/24

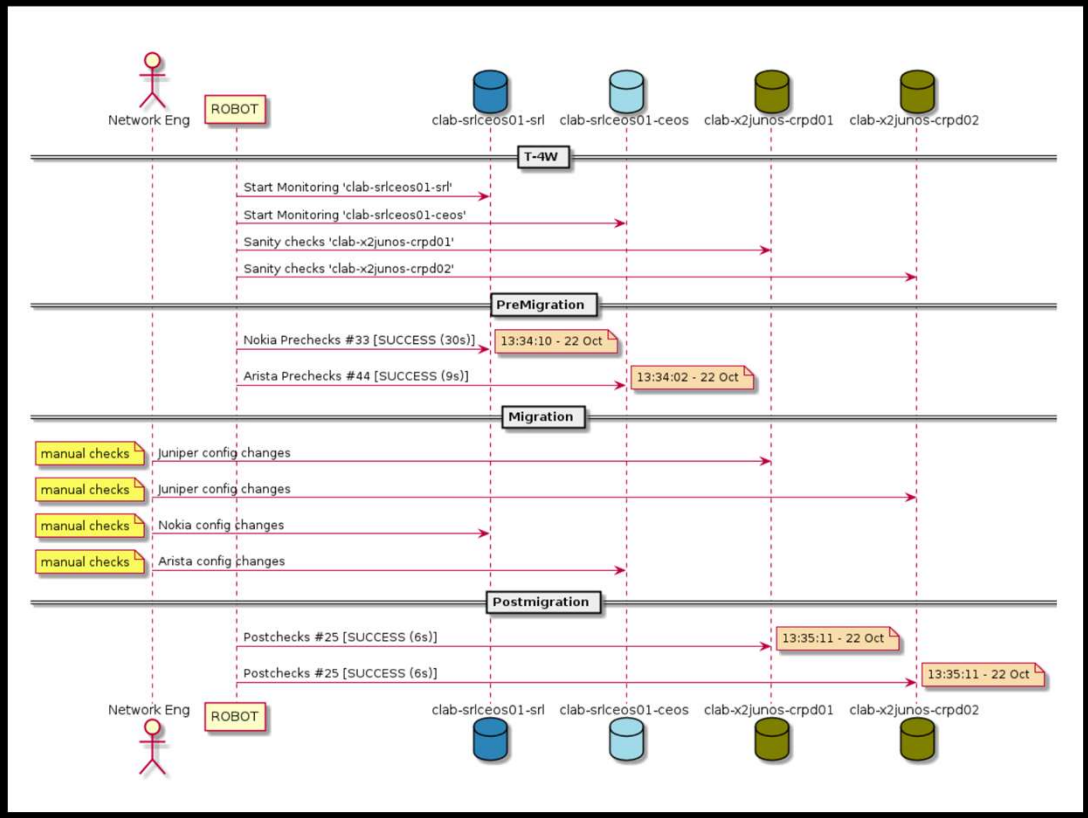
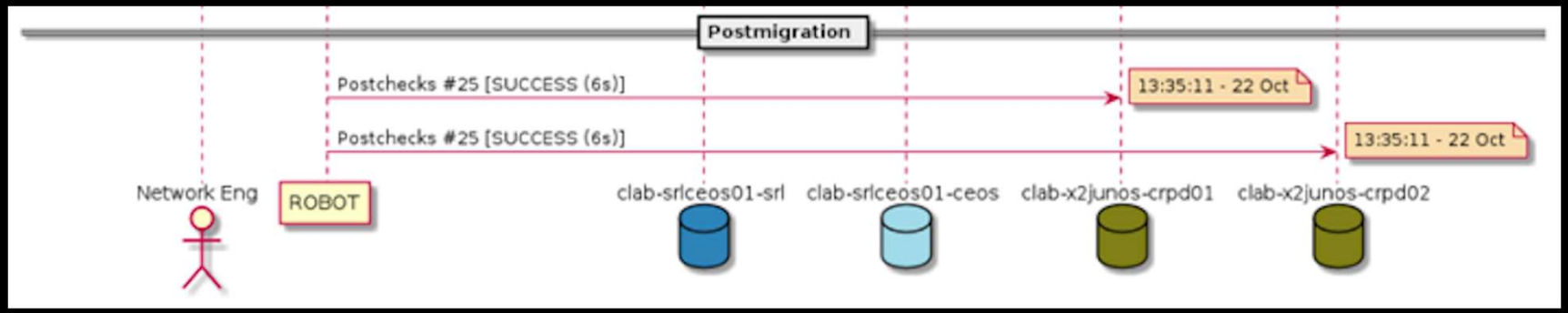
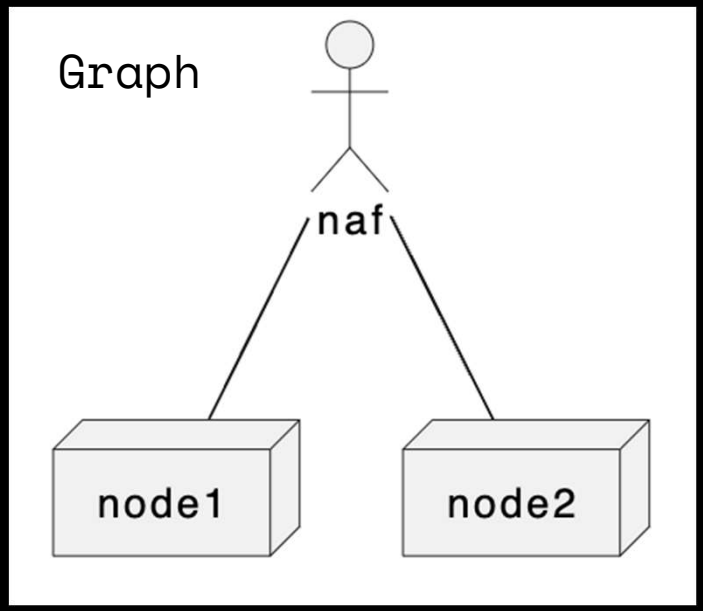
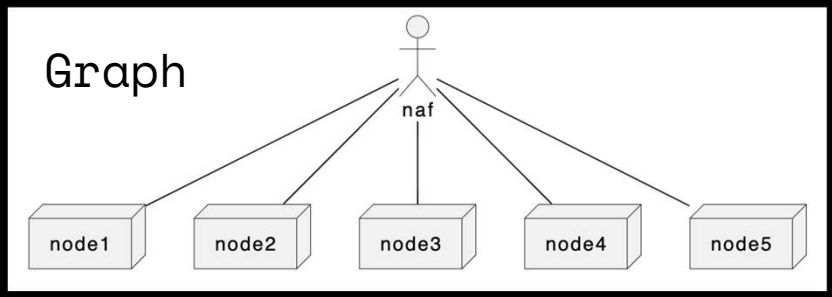
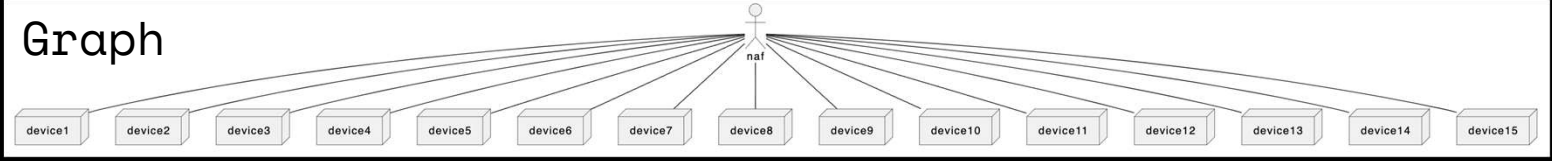
set interfaces ge-0/0/2 description "WAN interface"
set interfaces ge-0/0/2 unit 0 vlan-id 300
set interfaces ge-0/0/2 unit 0 family inet address 192.168.3.1/24
```

Jinja2 examples II

```
<interfaces>
{% for interface in interfaces -%}
    <interface>
        <name>{{ interface.name }}</name>
        <description>{{ interface.description }}</description>
        <unit>
        <id>{{ interface.unit }}</id>
        <vlan-id>{{ interface.vlan_id }}</vlan-id>
        <ipv4>{{ interface.ipv4_address }}</ipv4>
        {% if interface.ipv6_address -%}
        <ipv6>{{ interface.ipv6_address }}</ipv6>
        {% endif -%}
        </unit>
    </interface>
{% endfor -%}
</interfaces>
```

Jinja2 examples III

Jinja2 examples III

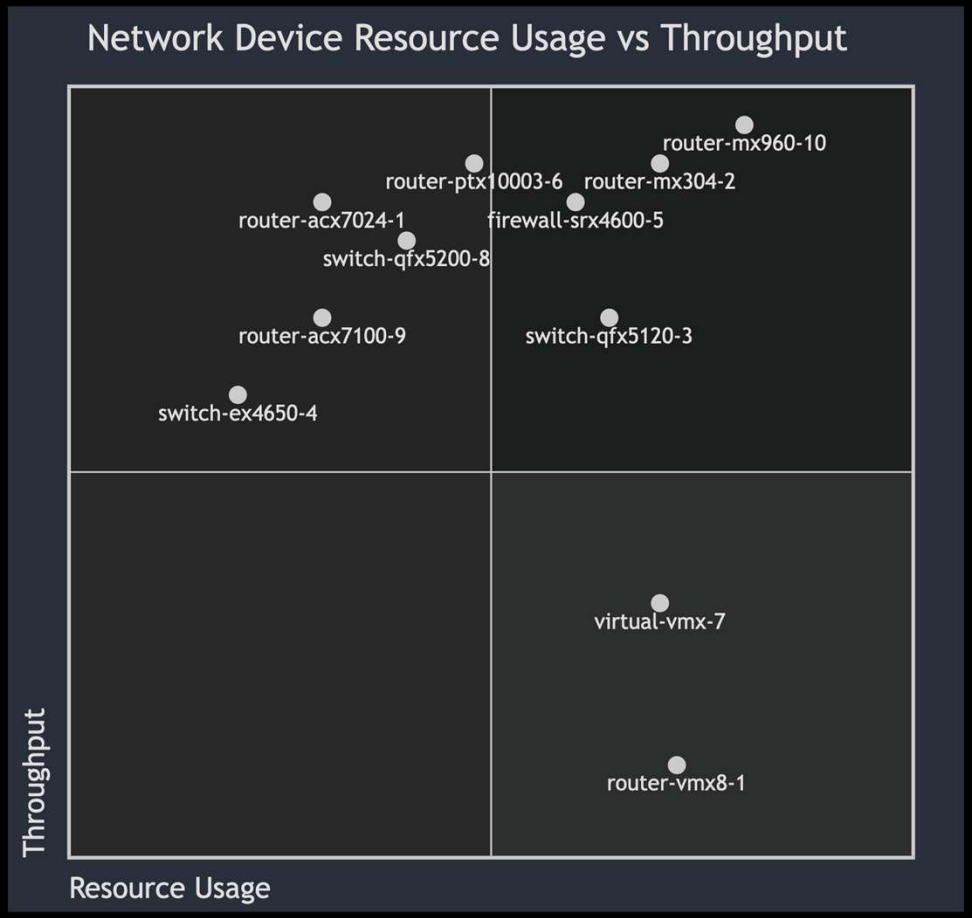
```
```plantuml
@startuml

actor "Network Eng" as ne
participant ROBOT
database "clab-srlceos01-srl"  #steelblue
database "clab-srlceos01-ceos" #lightblue
database "clab-x2junos-crpd01" #olive
database "clab-x2junos-crpd02" #olive

== T-4W ==

ROBOT -> "clab-srlceos01-srl": Start Monitoring 'clab-srlceos01-srl'
ROBOT -> "clab-srlceos01-ceos": Start Monitoring 'clab-srlceos01-ceos'
ROBOT -> "clab-x2junos-crpd01": Sanity checks 'clab-x2junos-crpd01'
ROBOT -> "clab-x2junos-crpd02": Sanity checks 'clab-x2junos-crpd02'

== PreMigration ==

ROBOT -> "clab-srlceos01-srl": Nokia Prechecks #{{ tcnsrl.matches.0.number }} [{{ tcnsrlj_result.matches.0.result }}]
note right #wheat: {{ '%H:%M:%S - %d %b' | strftime(tcnsrlj_timestamp.matches.0.timestamp[:-3]) }}
ROBOT -> "clab-srlceos01-ceos": Arista Prechecks #{{ tcnceos.matches.0.number }} [{{ tcnceosj_result.matches.0.result }}]
note right #wheat: {{ '%H:%M:%S - %d %b' | strftime(tcnceosj_timestamp.matches.0.timestamp[:-3]) }}

== Migration ==

ne -> "clab-x2junos-crpd01" : Juniper config changes
note left: manual checks
ne -> "clab-x2junos-crpd02" : Juniper config changes
note left: manual checks
ne -> "clab-srlceos01-srl" : Nokia config changes
note left: manual checks
ne -> "clab-srlceos01-ceos" : Arista config changes
note left: manual checks

== Postmigration ==

ROBOT -> "clab-x2junos-crpd01": Postchecks #{{ tcnjnpr1.matches.0.number }} [{{ tcnjnprj1_result.matches.0.result }}]
note right #wheat: {{ '%H:%M:%S - %d %b' | strftime(tcnjnprj_timestamp.matches.0.timestamp[:-3]) }}

ROBOT -> "clab-x2junos-crpd02": Postchecks #{{ tcnjnpr2.matches.0.number }} [{{ tcnjnprj2_result.matches.0.result }}]
note right #wheat: {{ '%H:%M:%S - %d %b' | strftime(tcnjnprj_timestamp.matches.0.timestamp[:-3]) }}

@enduml
```
```



Jinja2 examples III - zoom

```
36  == Postmigration ==
37
38  ROBOT -> "clab-x2junos-crpd01": Postchecks #{{ tcnjnpr1.matches.0.number }} [{{ tcnjnprj1_result.matches.0.result }}]
39  note right #wheat: {{ '%H:%M:%S - %d %b' | strftime(tcnjnprj_timestamp.matches.0.timestamp[:-3]) }}
40
41  ROBOT -> "clab-x2junos-crpd02": Postchecks #{{ tcnjnpr2.matches.0.number }} [{{ tcnjnprj2_result.matches.0.result }}]
42  note right #wheat: {{ '%H:%M:%S - %d %b' | strftime(tcnjnprj_timestamp.matches.0.timestamp[:-3]) }}
43
44  @enduml
45  ```
```



Jinja2 examples III

Data

```
actor_name = "naf"
nodes = ["node1", "node2"]
```

{ 1x actor
{ 2x nodes

Template

```
```plantuml
@startuml
actor {{ actor_name }}
{% for node in nodes -%}
node {{ node }}
{% endfor %}
{% for node in nodes -%}
{{ actor_name }} -- {{ node }}
{% endfor -%}
@enduml
```
```

Code

```
```plantuml
@startuml
actor naf
node node1
node node2

naf -- node1
naf -- node2
@enduml
```
```

Graph



Jinja2 examples III

Data

```
actor_name = "naf"
nodes = ["node{}".format(i) for i in range(1, 6)]
```

{ 1x actor
{ 5x nodes

Template

```
```plantuml
@startuml
actor {{ actor_name }}
{% for node in nodes -%}
node {{ node }}
{% endfor %}
{% for node in nodes -%}
{{ actor_name }} -- {{ node }}
{% endfor -%}
@enduml
```
```

Code

```
```plantuml
@startuml
actor naf
node node1
node node2
node node3
node node4
node node5

naf -- node1
naf -- node2
naf -- node3
naf -- node4
naf -- node5
@enduml
```
```

Graph



Jinja2 examples III

Graph



Template

```
```plantuml
@startuml
actor {{ actor_name }}
{% for node in nodes -%}
node {{ node }}
{% endfor %}
{% for node in nodes -%}
{{ actor_name }} -- {{ node }}
{% endfor -%}
@enduml
```
```

Data

```
actor_name = "naf"
nodes = ["device{}".format(i) for i in range(1, 16)]
```

{ 1x actor

15x nodes

Code

```
```plantuml
@startuml
actor naf
node device1
node device2
node device3
node device4
node device5
node device6
node device7
node device8
node device9
node device10
node device11
node device12
node device13
node device14
node device15

naf -- device1
naf -- device2
naf -- device3
naf -- device4
naf -- device5
naf -- device6
naf -- device7
naf -- device8
naf -- device9
naf -- device10
naf -- device11
naf -- device12
naf -- device13
naf -- device14
naf -- device15
@enduml
```
```

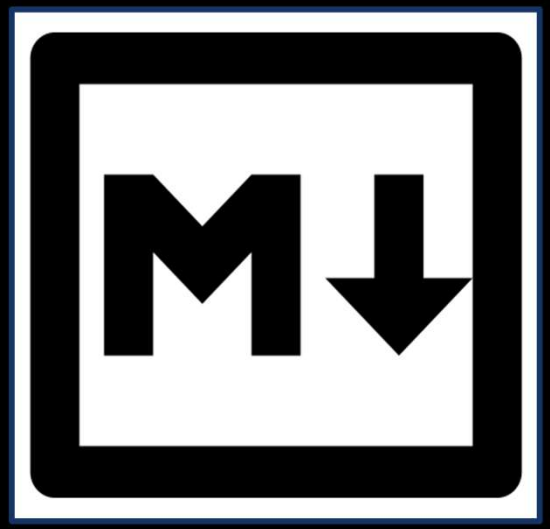
Jinja2 examples IV

Jinja2 examples IV

```mermaid
quadrantChart
    title {{ title }}
    x-axis {{ x_axis }}
    y-axis {{ y_axis }}
{% for device, coords in devices.items() -%}
    "{{ device }}": [{{ coords[0] }}, {{ coords[1] }}]
{% endfor -%}
```



Jinja2 examples V



Jinja2 examples V

```
# {{ title }}

**Author**: {{ author }}

---

{% for section in sections %}
## {{ section.name }}

{{ section.description }}

{% if section.items %}
### Items:

| Name | Description |
| ---- | ----------- |
{% for item in section.items %}
| **{{ item.name }}** | {{ item.description }} |
{% endfor %}
{% endif %}
{% endfor %}

```

Network Configuration Report

Author: John Doe

System Information

Basic system info of the network device.

Items:

Name	Description
Hostname	MX960-Router
OS Version	JunOS 20.4R3

Interfaces

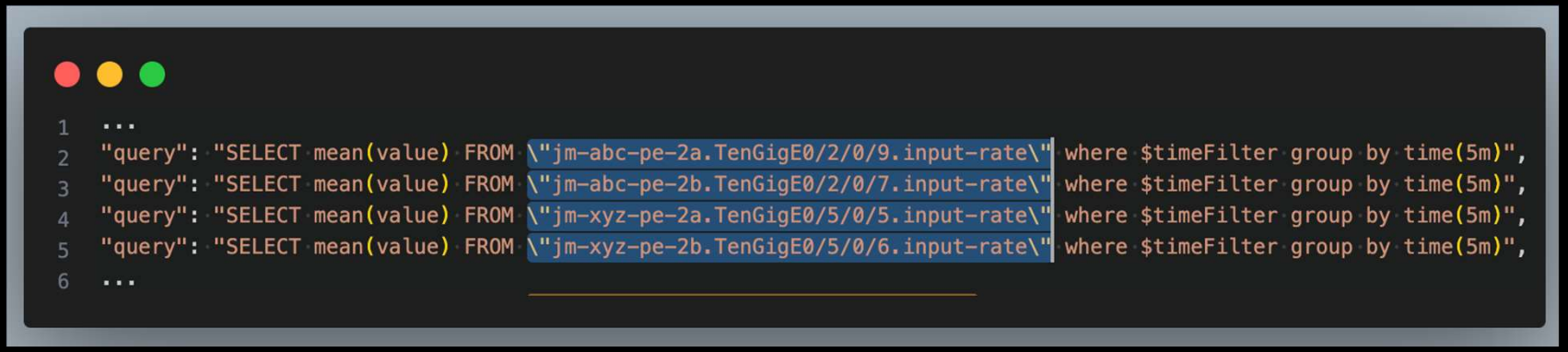
Overview of configured interfaces.

Items:

Name	Description
ge-0/0/0	Uplink to Core - 192.168.1.1/24
ge-0/0/1	LAN Interface - 192.168.2.1/24

Jinja2 examples VI

Jinja2 examples VI



```
...
"query": "SELECT mean(value) FROM \"jm-abc-pe-2a.TenGigE0/2/0/9.input-rate\" where $timeFilter group by time(5m)",
"query": "SELECT mean(value) FROM \"jm-abc-pe-2b.TenGigE0/2/0/7.input-rate\" where $timeFilter group by time(5m)",
"query": "SELECT mean(value) FROM \"jm-xyz-pe-2a.TenGigE0/5/0/5.input-rate\" where $timeFilter group by time(5m)",
"query": "SELECT mean(value) FROM \"jm-xyz-pe-2b.TenGigE0/5/0/6.input-rate\" where $timeFilter group by time(5m)",
...
```

```
...
"query": "SELECT mean(value) FROM \"{{ sd }}.{{ s_port }}.input-rate\" where $timeFilter group by time(5m)",
...
```

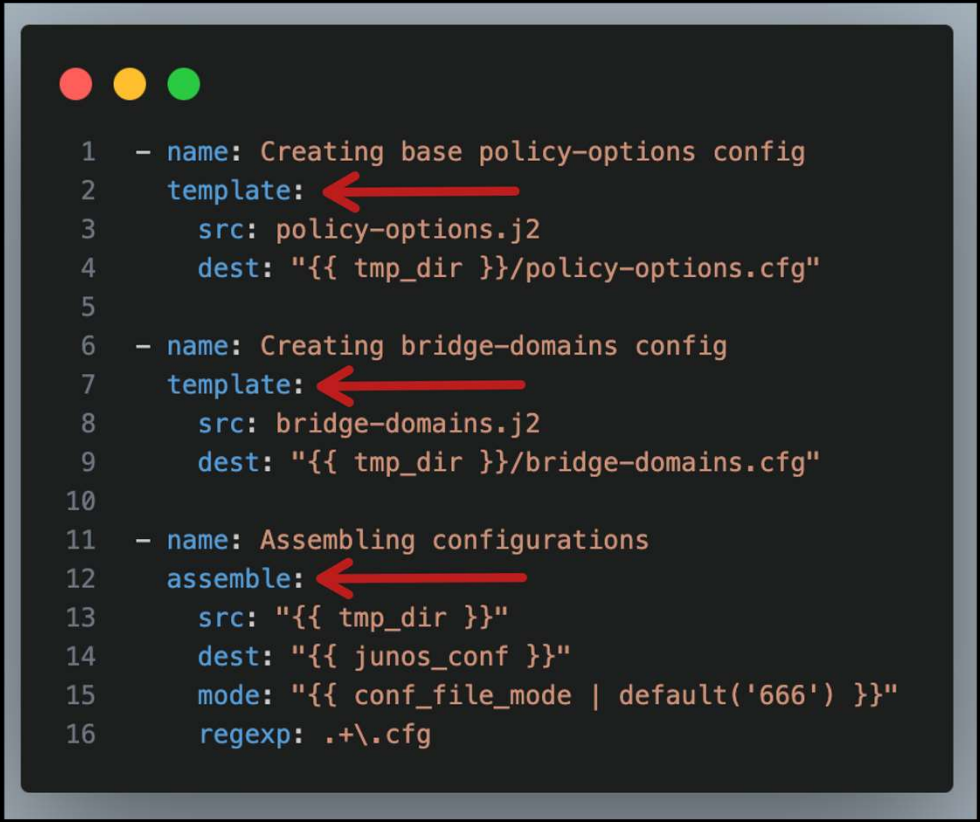
Ansible

- Automation tool
- Human readable (YAML syntax)
- Works great with jinja2 templates
- Easily integrated
- Built-in modules
- Extensible
- Cross-platform

Ansible examples I

```yaml
- name: Retrieve sanity_checks job last build (#{{ sanity_checks_jlbn.matches.0.number }}) details.
  uri:
    url: https://{{ jenkins.user }}:{{ jenkins.password }}@{{ jenkins.host }}:{{ jenkins.port }}/job/sanity_checks/{{ sanity_checks_jlbn.matches.0.number }}/api/xml? tree
    method: GET
    force_basic_auth: yes
    validate_certs: no
    status_code: 200
    return_content: true
  register: sanity_checks_job_details
  tags: sanity_checks

- name: Parse sanity_checks job last build duration.
  xml:
    xmlstring: "{{ sanity_checks_job_details.content }}"
    xpath: //duration
    content: text
  register: sanity_checks_job_duration
  tags: sanity_checks

- name: Parse sanity_checks job last build result.
  xml:
    xmlstring: "{{ sanity_checks_job_details.content }}"
    xpath: //result
    content: text
  register: sanity_checks_job_result
  tags: sanity_checks
```

Ansible examples II



```yaml
- name: Creating base policy-options config
  template:
    src: policy-options.j2
    dest: "{{ tmp_dir }}/policy-options.cfg"

- name: Creating bridge-domains config
  template:
    src: bridge-domains.j2
    dest: "{{ tmp_dir }}/bridge-domains.cfg"

- name: Assembling configurations
  assemble:
    src: "{{ tmp_dir }}"
    dest: "{{ junos_conf }}"
    mode: "{{ conf_file_mode | default('666') }}"
    regexp: .+\.cfg
```

Docker

- Containerization platform
- Isolated environments (containers)
- Portability
- Efficiency
- Consistency
- Dockerfile
- DevOps

Jenkins

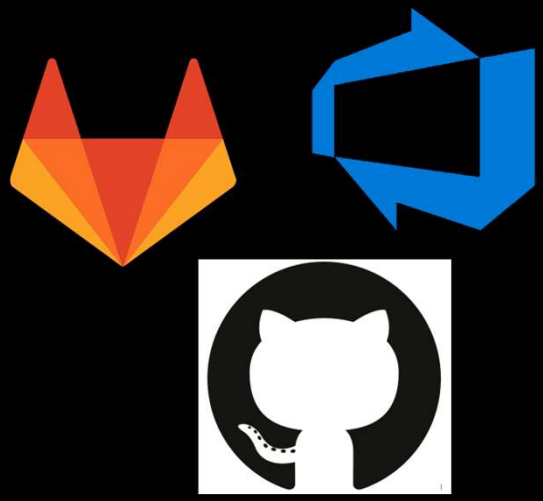
- Automation server
- Aprox. ~1800 plugins (robotframework, slack, gitlab, etc…)
- Customizability
- Jobs (pipelines, tasks, scripts…)
- DevOps

Git

- Track changes in files and directories
- Compare changes
- Different branches
- Commits
- Write your project own story

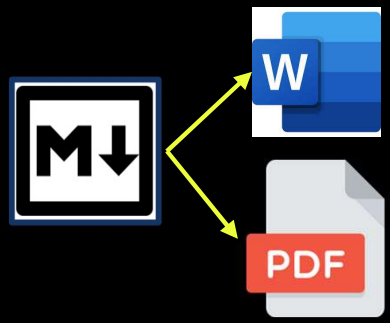
Version Control Systems

- DevOps platforms (GitLab, GitHub, Azure DevOps…)
- Git repo management
- Easily track changes (GUI)
- Easy to collaborate
- Change Management
- CI/CD
- Seamless Markdown integration



Pandoc

- Universal document converter (from ⇔ to)
- Flexibility
- Customizable
- https://pandoc.org/



LaTeX

- High quality technical documents
- Consistency
- Typesetting:
  - layout
  - fonts
  - ToC
  - spacing
  - alignment
  - tables
  - figures
- https://www.latex-project.org

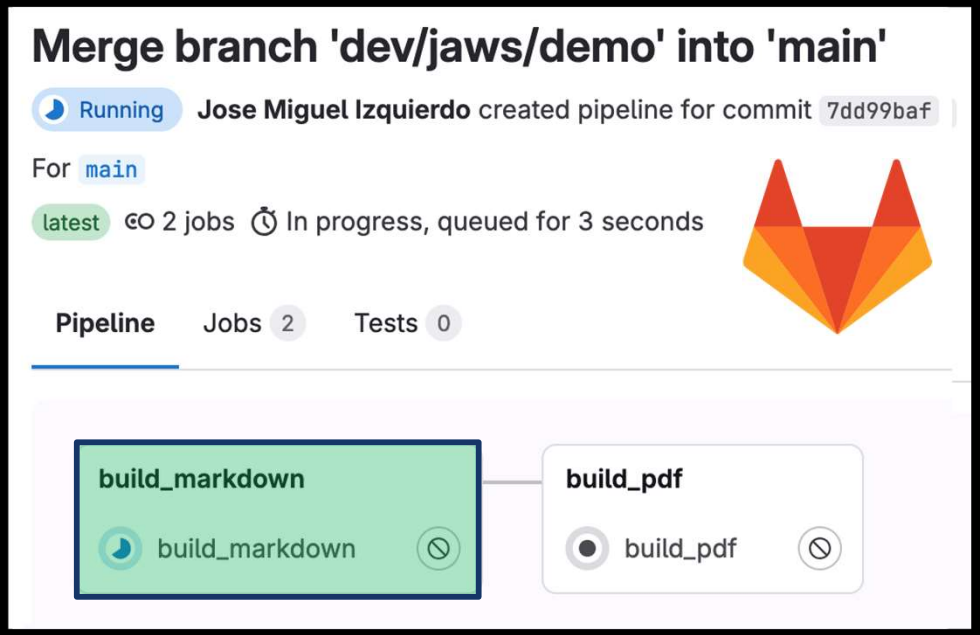
LATEX

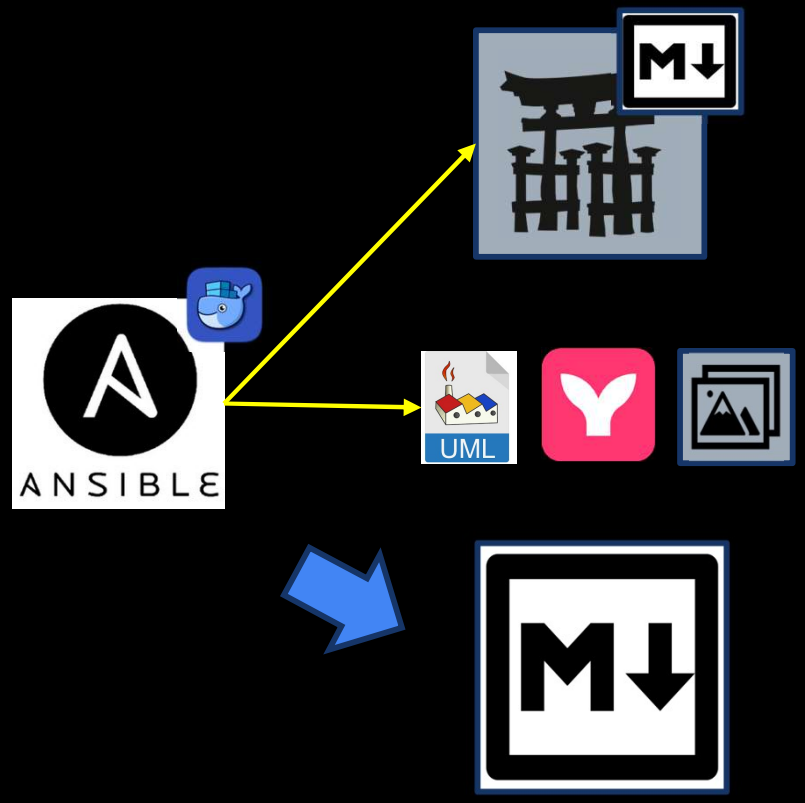

NAF
NETWORK AUTOMATION FORUM
Real World Use Case I
Product/Project documentation (PD)
Juniper Business Use Only

PD: Workflow

- *jinja2* templates (Markdown + code blocks)
- Graphs (static [Diagrams As Code] + static [pictures])
- Ansible (orchestrator)
- Data (mainly static: *group_vars*, *host_vars*)
- Triggered by a Merge request (MR)

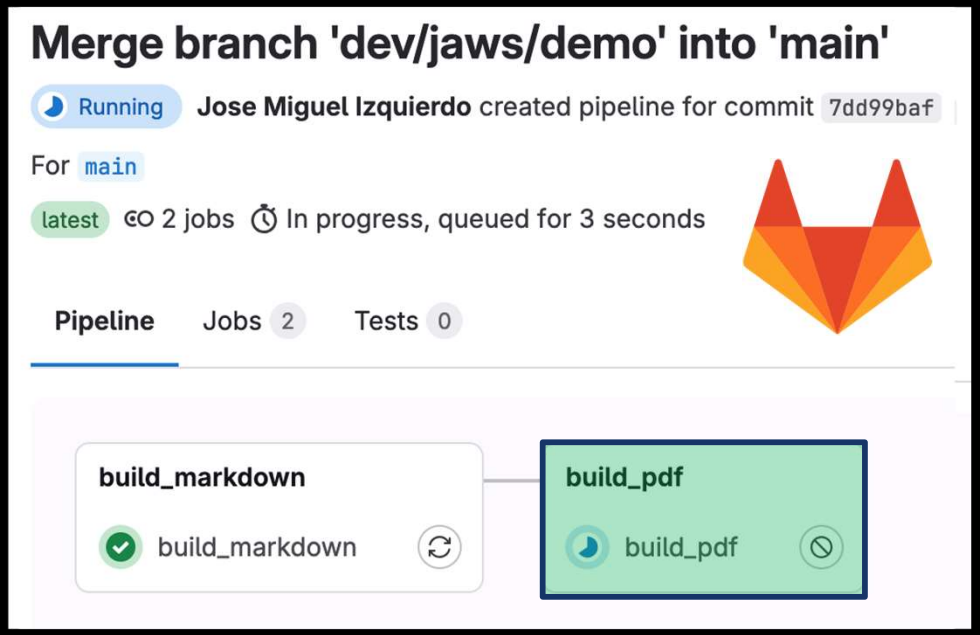
PD: Pipeline - Stage #1

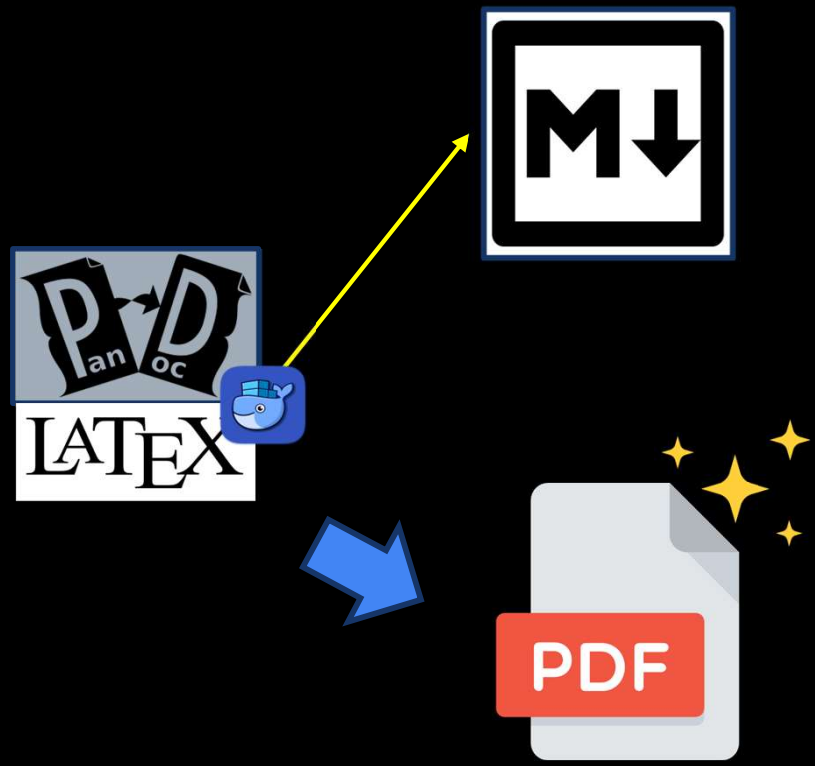




.gitlab-ci.yml

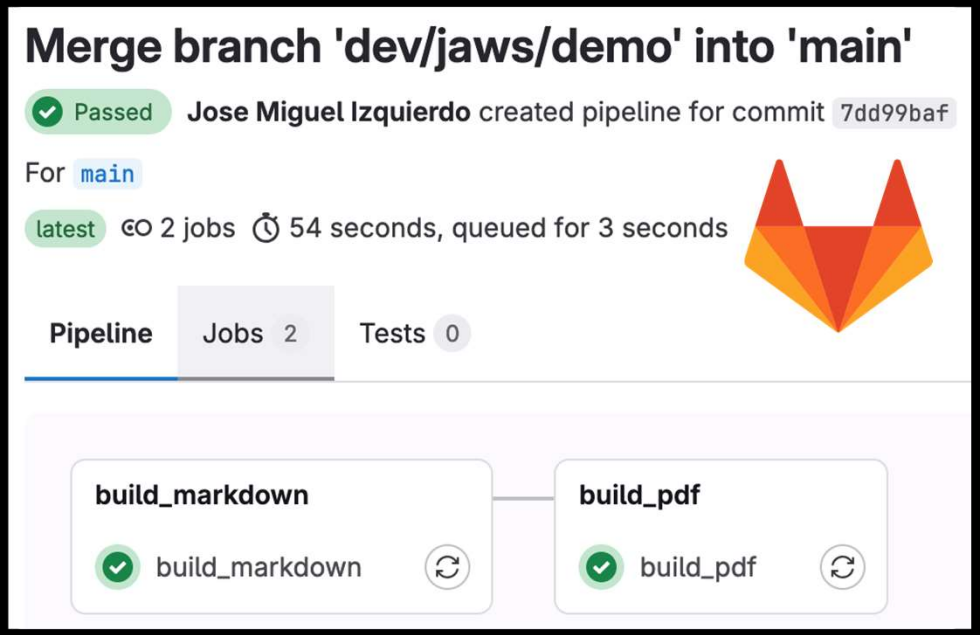
PD: Pipeline - Stage #2

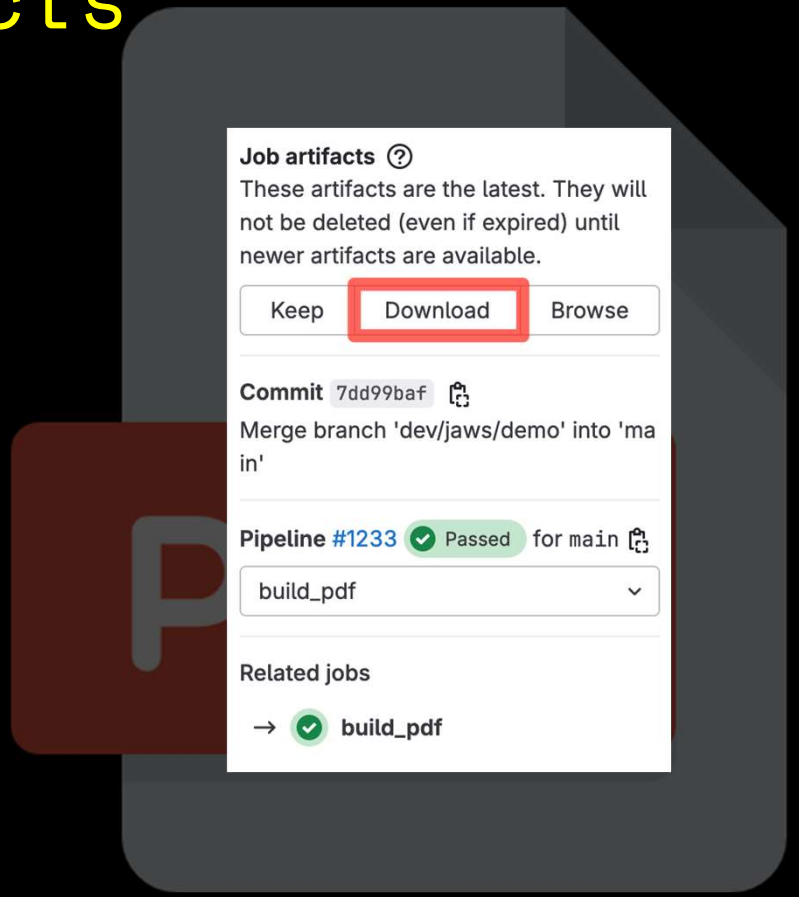




.gitlab-ci.yml

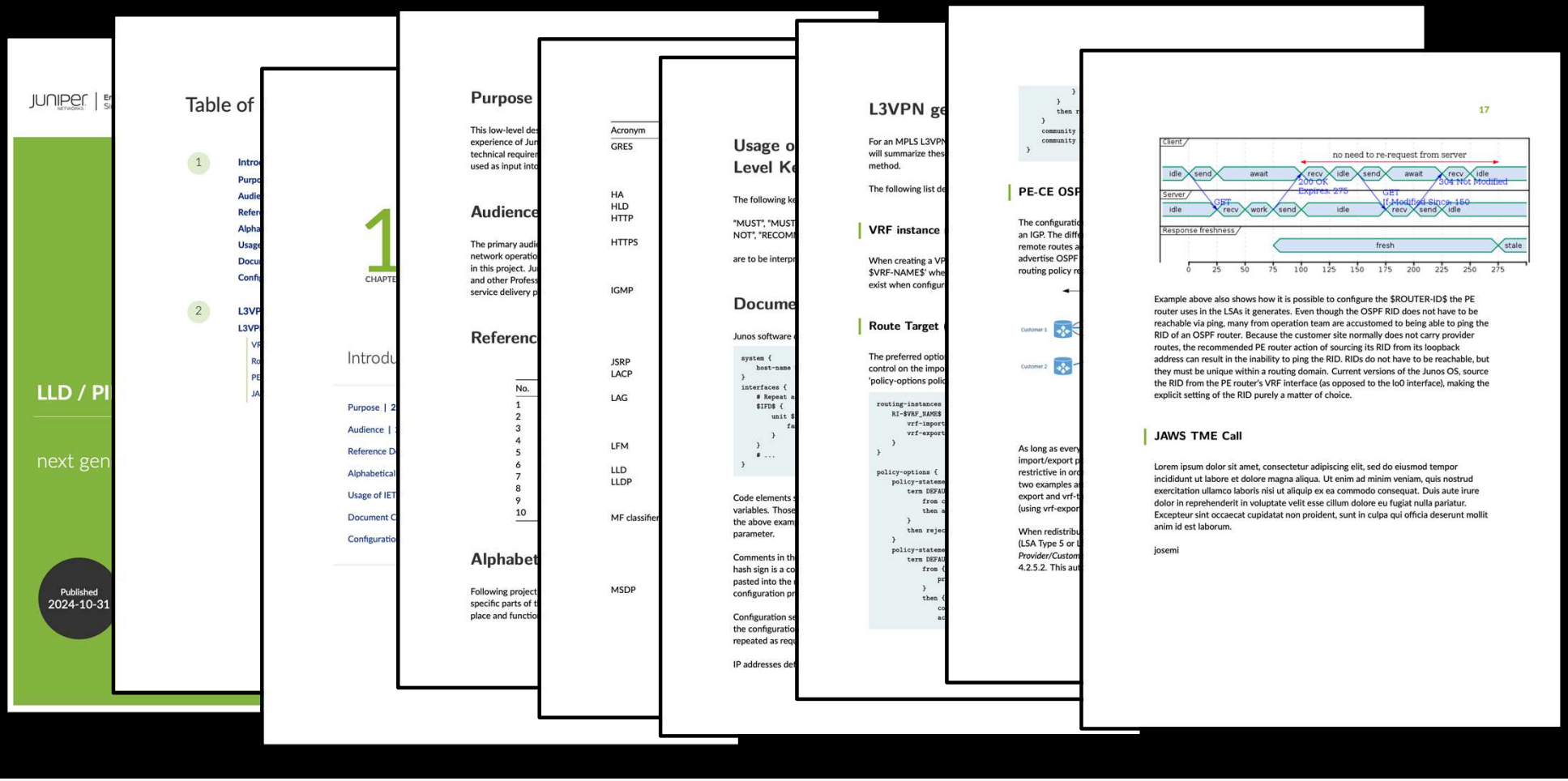
PD: Pipeline - Artifacts

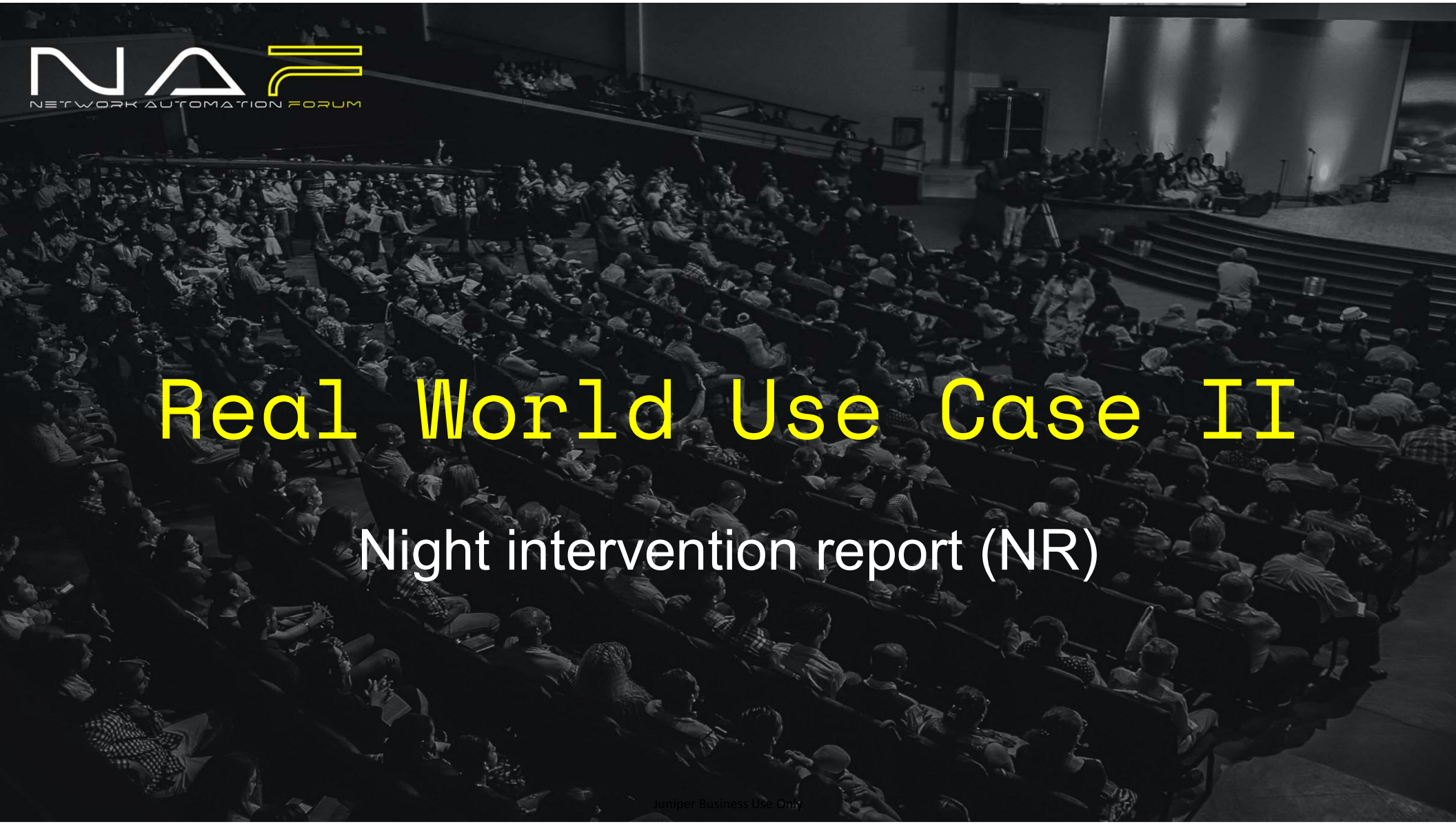




.gitlab-ci.yml

PD: Documentation

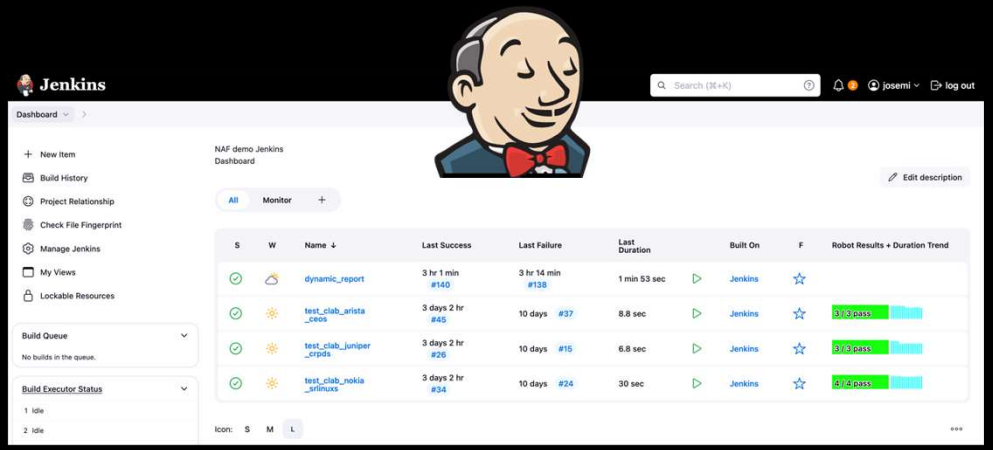


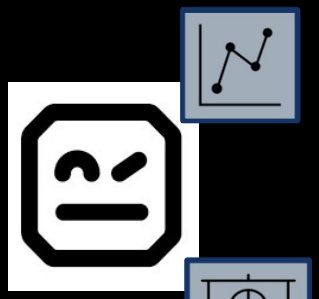

NAF
NETWORK AUTOMATION FORUM
Real World Use Case II
Night intervention report (NR)
Juniper Business Use Only

NR: Workflow

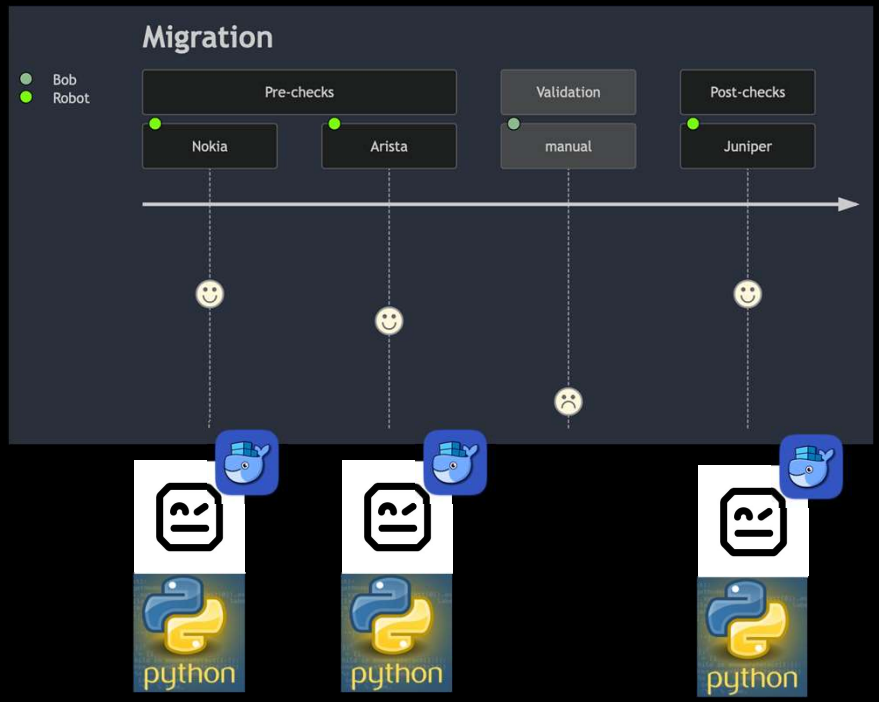
- *jinja2* templates (Markdown + code blocks)
- Graphs (dynamic [diagrams As Code] + static [pictures])
- Ansible (orchestrator)
- Data (3rd party systems: Grafana, Robotframework)
- Artifacts

NR: Tests

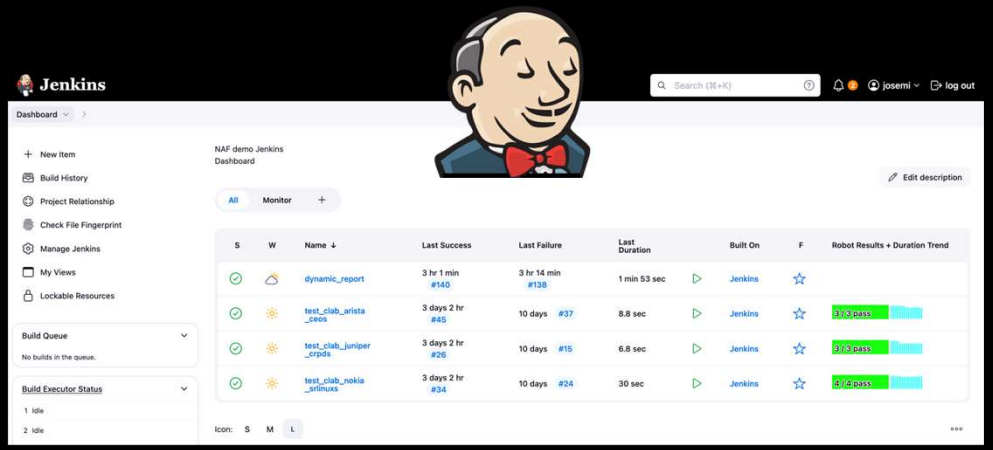


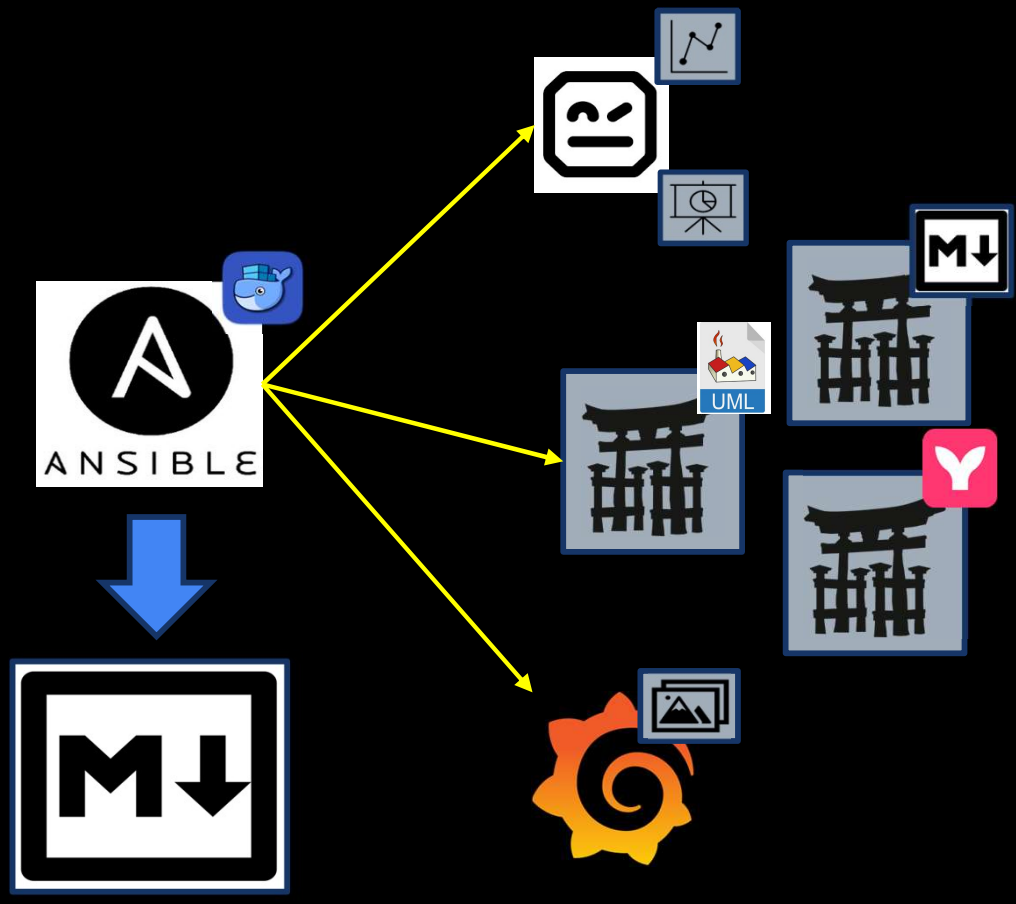


- Reports…
- Logs…
- Operational command outputs…

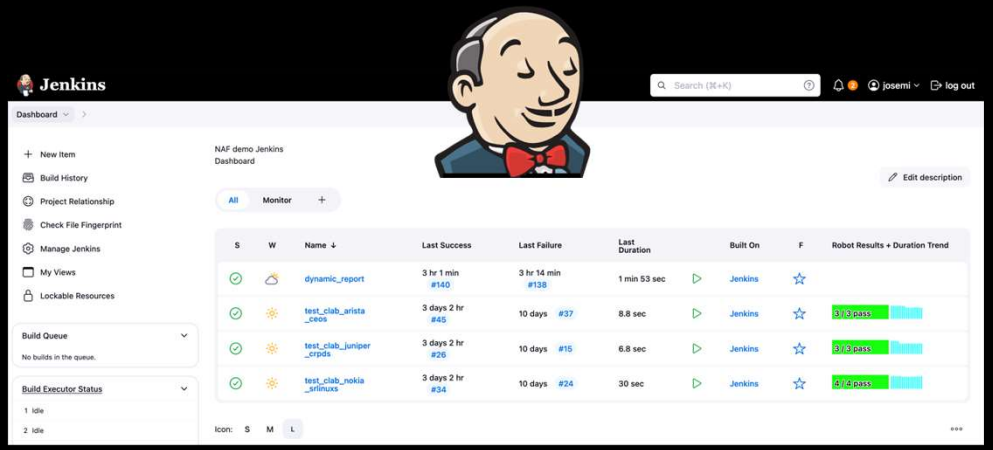


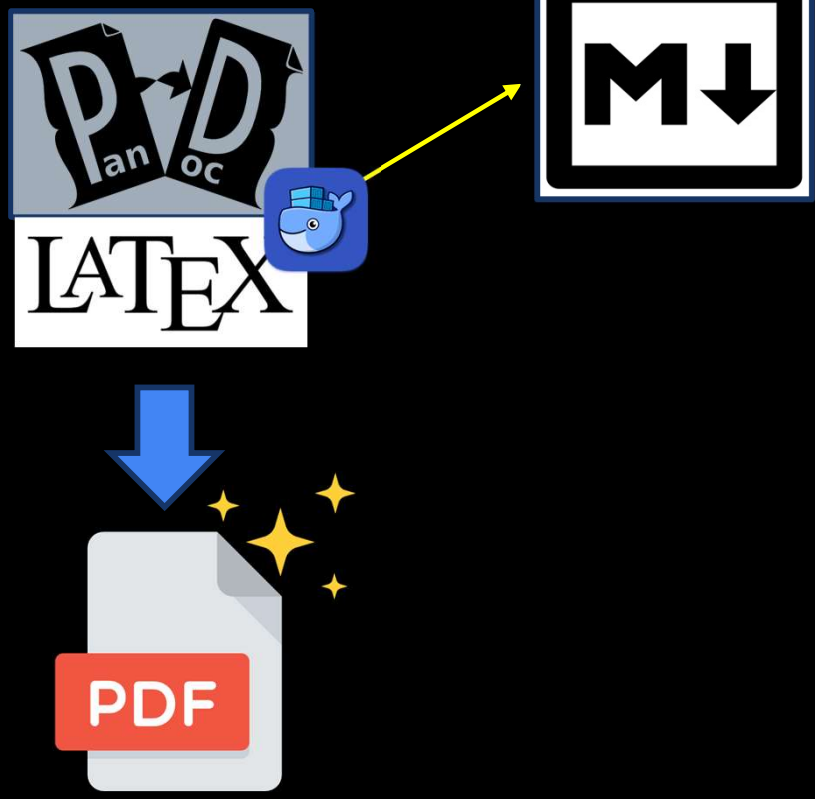
NR: Report


Jenkins
Search (⌘+K)
josemi
log out
Dashboard
New Item
Build History
Project Relationship
Check File Fingerprint
Manage Jenkins
My Views
Lockable Resources
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle
NAF demo Jenkins
Dashboard
Edit description
All
Monitor
S
W
Name ↓
Last Success
Last Failure
Last Duration
Built On
F
Robot Results + Duration Trend
dynamic_report
3 hr 1 min #140
3 hr 14 min #138
1 min 53 sec
Jenkins
test_clab_arista_ceos
3 days 2 hr #45
10 days #37
8.8 sec
Jenkins
3 / 3 pass
test_clab_juniper_crpds
3 days 2 hr #26
10 days #15
6.8 sec
Jenkins
3 / 3 pass
test_clab_nokia_srlinuxs
3 days 2 hr #34
10 days #24
30 sec
Jenkins
4 / 4 pass
Icon: S M L

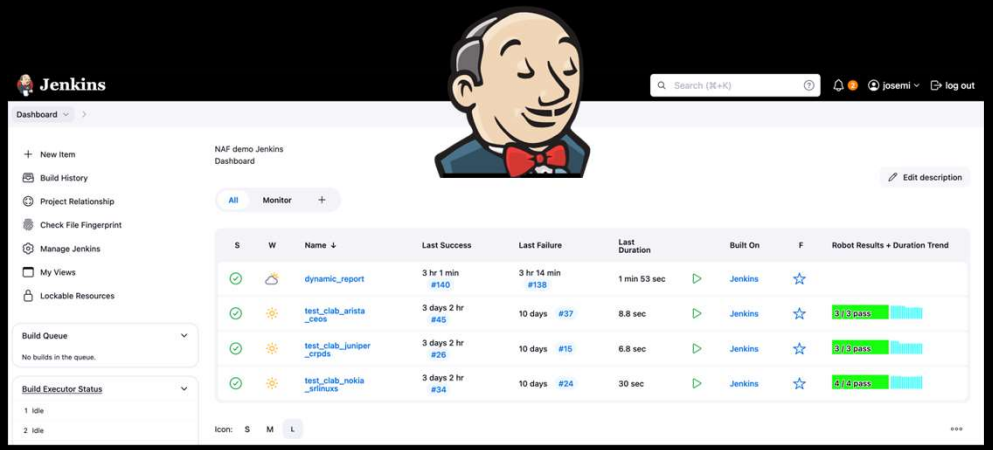

ANSIBLE
UML

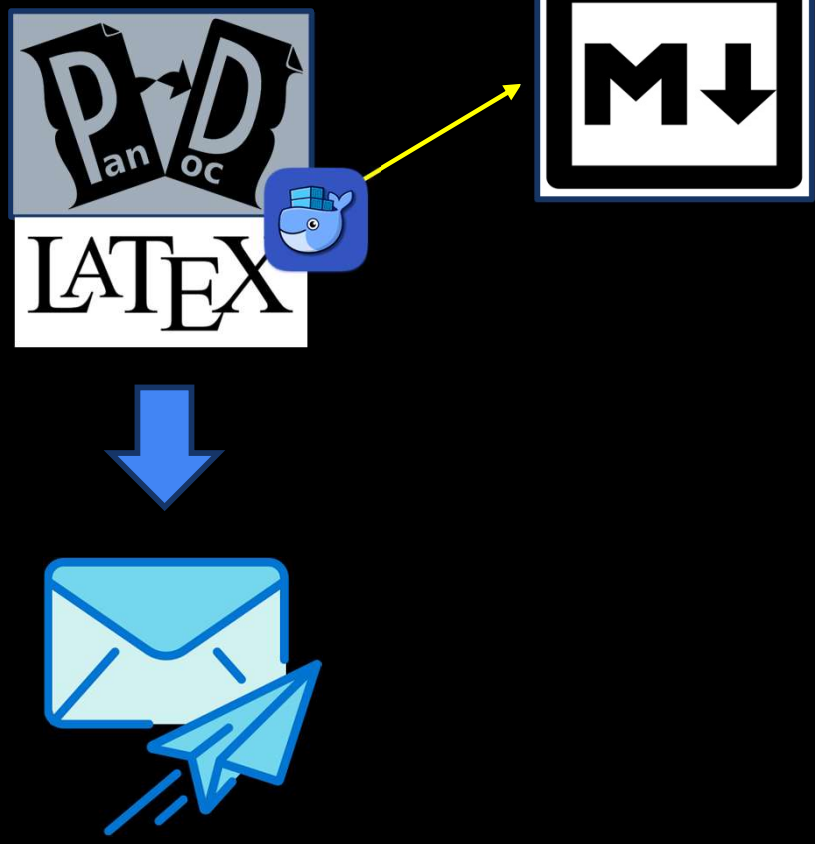
NR: PDF Report



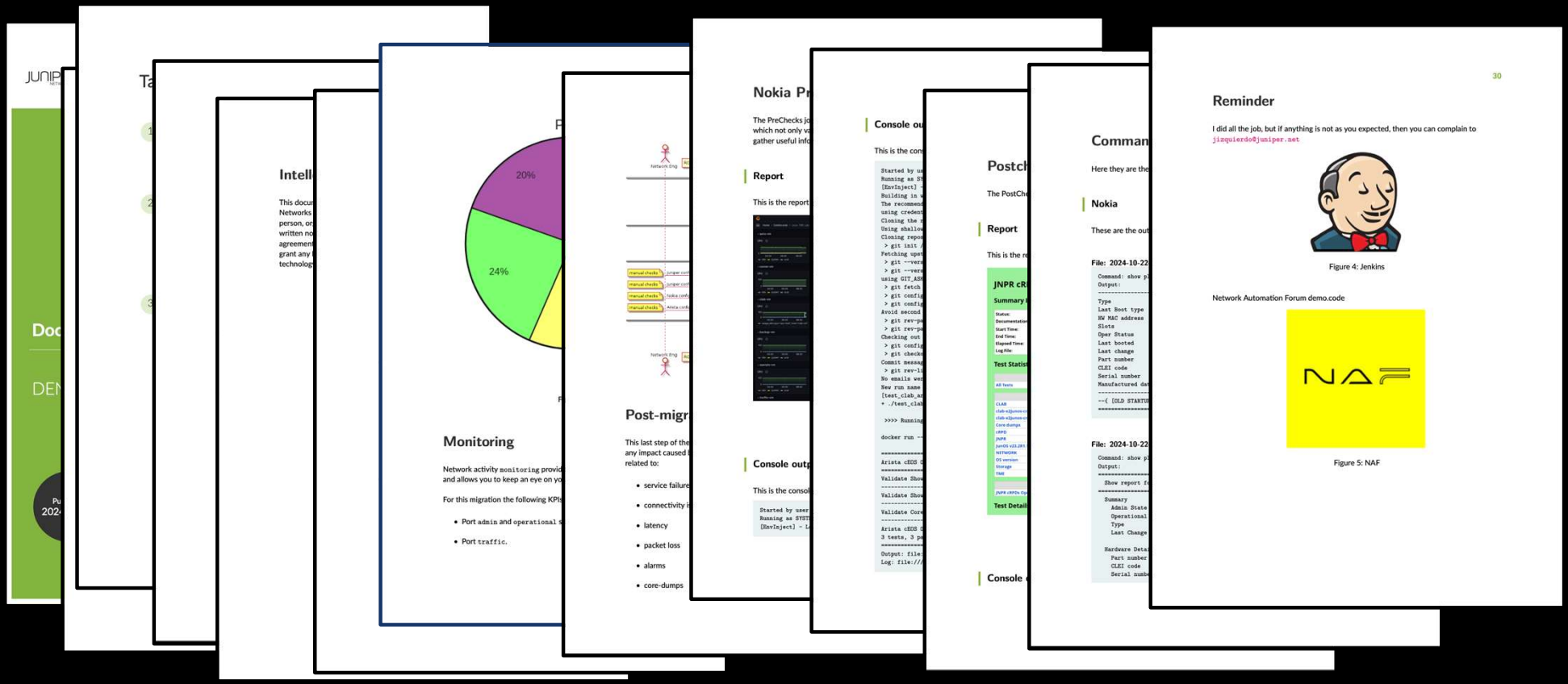


NR: PDF Report





NR: Documentation



Summary

- Identify your use case goals:
    - repeatable
    - consistent look & feel
    - etc…
- Select your tools (recommendation start with the first 5)
- No need to be a coding expert
- Define your workflow integrations

The end

Template

```
1
2  {% for job in cast_and_crew -%}
3    {{ "{:<16}".format(job) }} - Jose Miguel Izquierdo
4  {% endfor -%}
5
6  {% if thanks %}
7  Special thanks to:
8  {% for ack in thanks.acknowledgements %}
9  - {{ ack }}!
10 {% endfor %}
11 All - {{ thanks.msg }}
12 {% endif -%}
13
```

Data

```
1  cast_and_crew:
2    - "Casting"
3    - "Slides writer"
4    - "Designer"
5    - "Makeup"
6    - "Production"
7    - "Presenter"
8
9  thanks:
10   acknowledgements:
11     - "Frank Reimer"
12   msg: "THANK YOU VERY MUCH!!!!"
```

The end

Code

```
Casting          - Jose Miguel Izquierdo
Slides writer    - Jose Miguel Izquierdo
Designer         - Jose Miguel Izquierdo
Makeup           - Jose Miguel Izquierdo
Production       - Jose Miguel Izquierdo
Presenter        - Jose Miguel Izquierdo

Special thanks to:

- Frank Reimer!

All - THANK YOU VERY MUCH!!!!
```