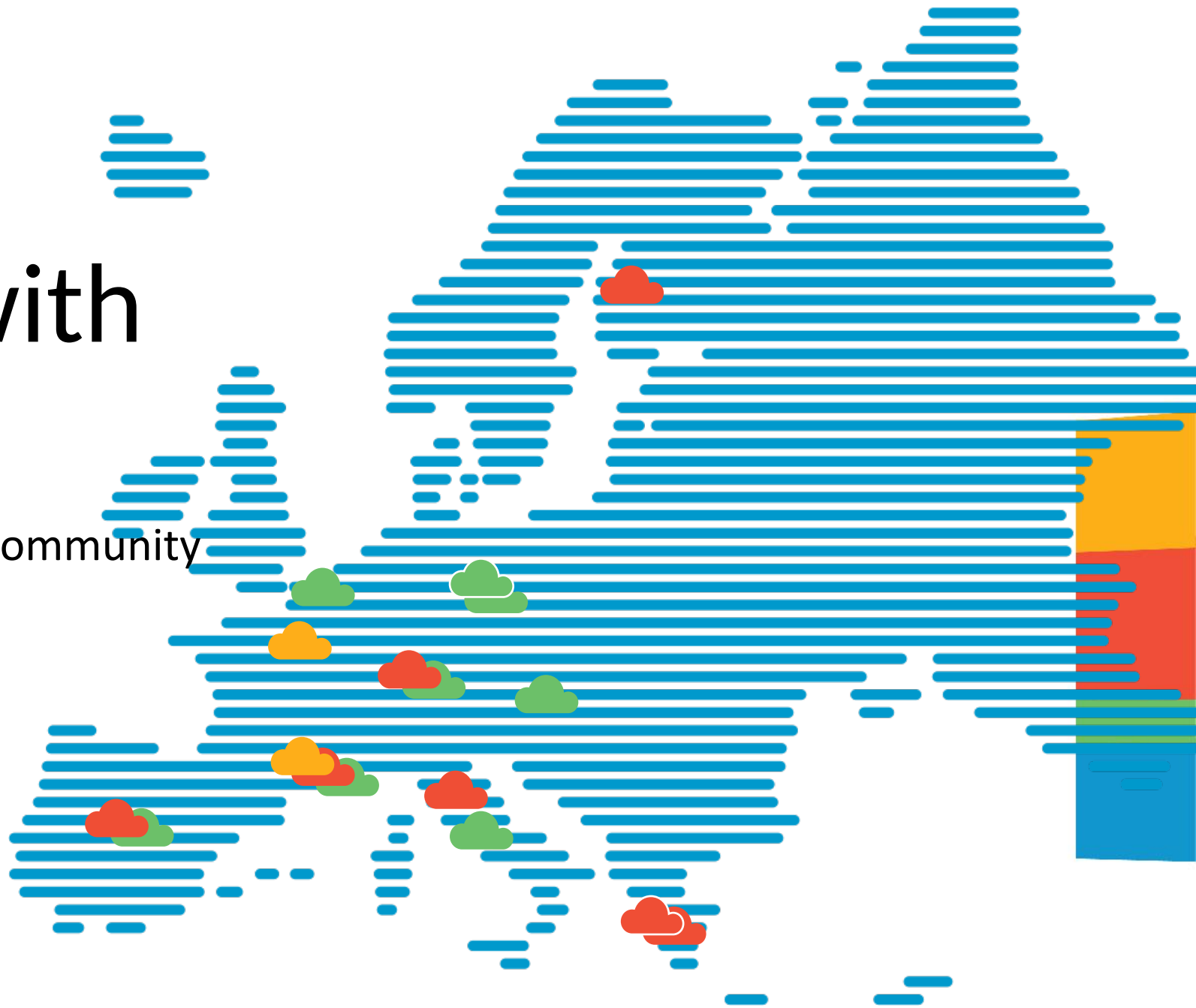




Rethink testbeds with blueprints

on behalf of the SLICES community

11/30/2023



core



edge



RAN

The blueprint concept in a nutshell

- Help collaboration between engineers and non-engineers to collaborate on developing (business) applications.
- Define a common terminology that doesn't require in-depth, technical knowledge
 - to produce consistent vision on the application with a focus on the future objective.
- Pick up the right resources, be iterative, review and validate, and keep it as a baseline.

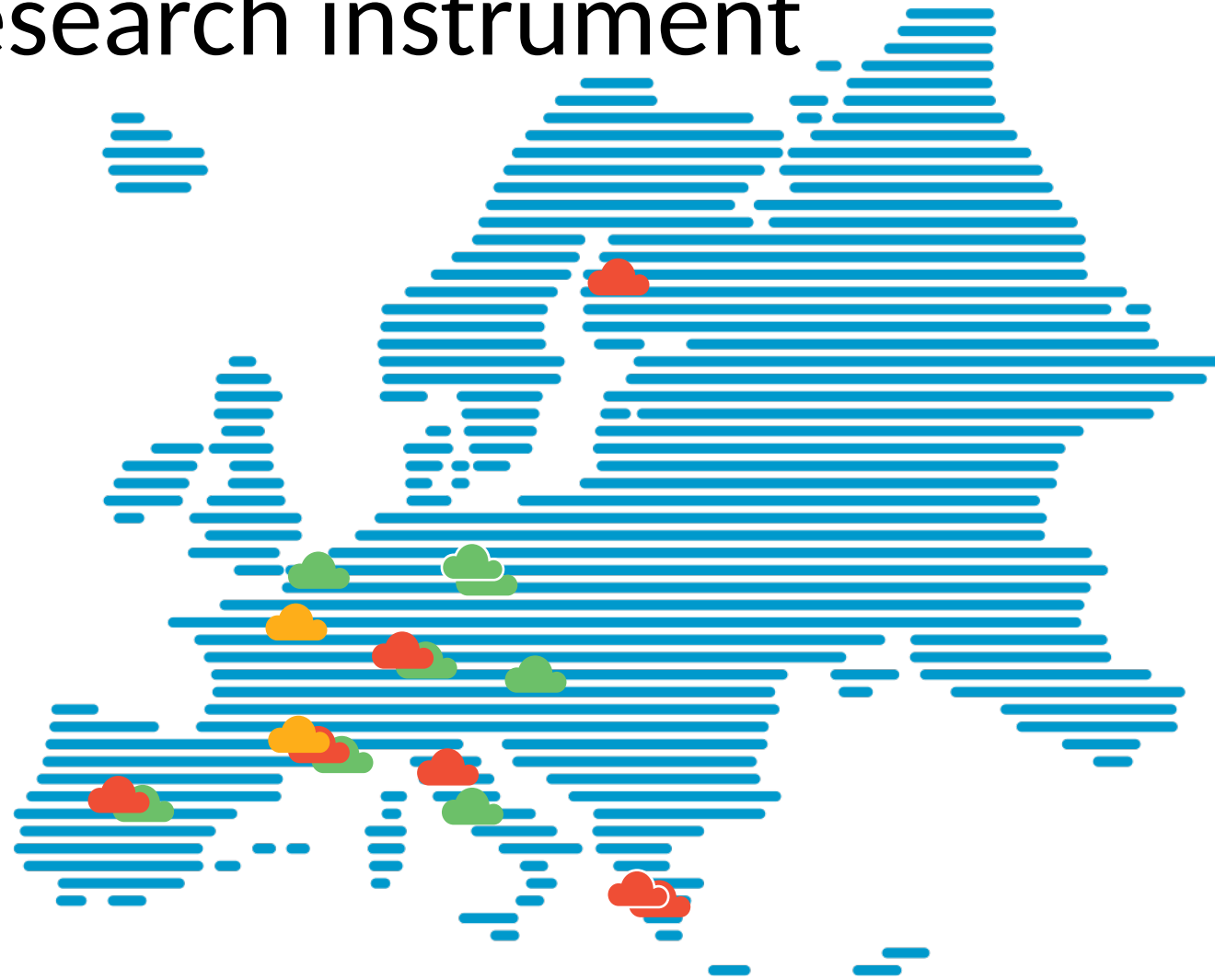
Adapted from [0a, 0b]



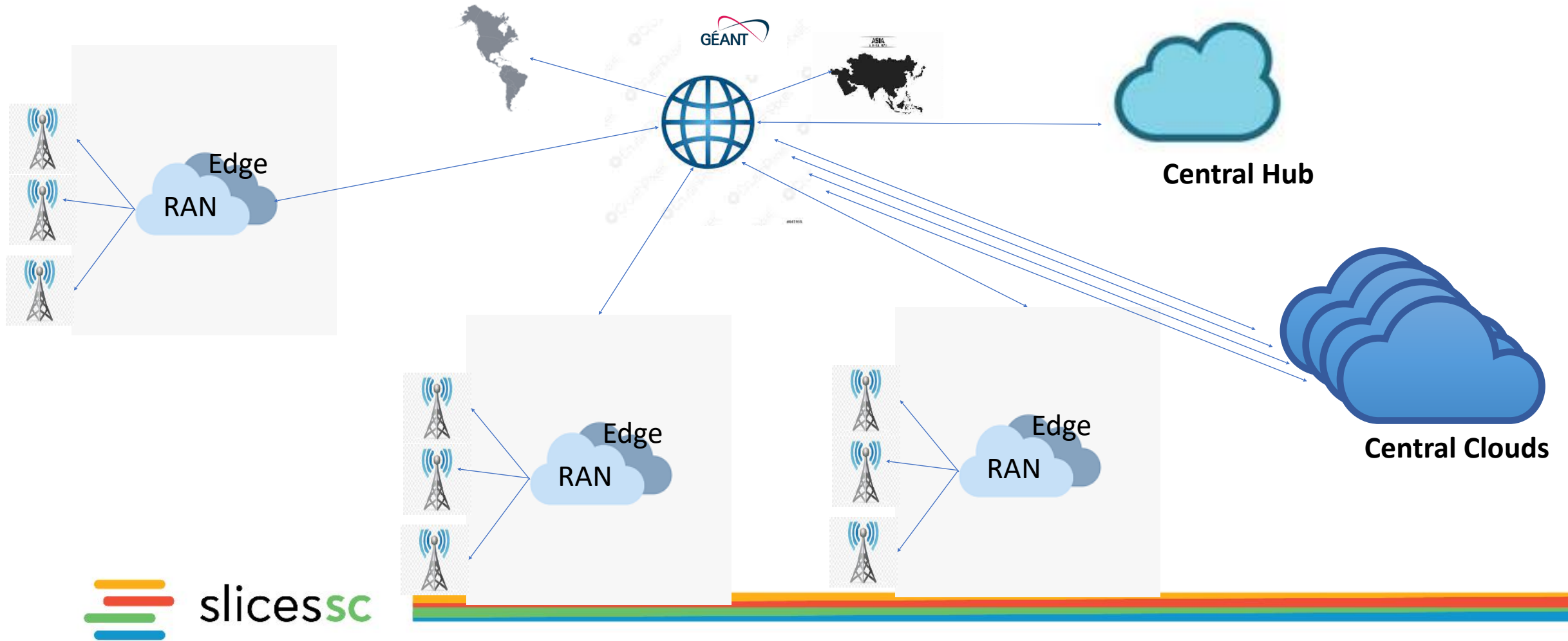
[0a] <https://dev.to/jayjayjpg/what-is-a-software-blueprint-5388>

[0b] <https://www.qrpinternational.fr/blog/faq/blueprint-quest-ce-que-cest/>

SLICES research instrument



5G blueprint – High level view



Open, large scale, reproducible

- Reuse and contribute to open-source initiatives
 - Common software/hardware base
- Complex deployments:
 - Multi-region
 - Multi-tenancy
 - Multi-management
- Full documentation
- Fine-grain automatic control

What for?



Type 1 experiments: vertical service integration and testing

- post-5G system is a blackbox and users embed software
 - a) in the network infrastructure user-plane, either in the a1) cloud center or a2) at the RAN/Edge (low-latency services)
 - b) terminals (e.g. drones, robots, fixed stations).
- Users can add HW (“bring your own device”) to the terminal end for special experiments (e.g. multimedia or connected robotics). Potentially also in the network infrastructure.

Type 2 experiments: Software Defined Networking

- The user has access to well-defined interfaces to the network functions (e.g., xAPP) and writes applications which stimulate the interface to alter the network behaviour or collect KPI.
- This typically involves writing software in the framework of a controller.
- Eventually it could even be in a DPU or programmable switches (P4) for low-level real-time behaviour.

Type 3 experiments: radio/network development software

- Users without specific radio/processing HW can use a site for development and testing purposes (e.g., ssh access) with the objective of pushing software improvements to companion OSS projects.
- A user develops improved network function (e.g., PHY, MAC) implementations using one of the CI/CD frameworks of companion projects (e.g. OAI, ORAN O-SC) used by SLICES-RI sites for CD. These go through the normal CI procedures (testing) and get deployed on SLICES-RI for larger-scale testing.

Type 4 experiments: low-level access to radio resources

- Will be to experiment with candidate 6G technologies. This will typically involve insertion of hardware elements into sites such as
 - novel RF devices such as antennas, RIS, optical wireless devices, THz radios, etc.
 - hardware accelerators for key functions like channel decoders, cryptographic functions, user-plane packet-processing.
 - real-time edge devices (TSN, real-time multimedia, industrial IoT, etc.).
- Once inserted (or provided by sites) users can reserve time to develop and test scenarios using these devices.

Type 5 experiments: joint use of post-5G infrastructure and HPC resources

- Example 1: Real-time Digital twin of radio network
 - GPU farms can be used as real-time 3D radio emulators. When interconnected with radio and core network infrastructure can make a digital twin of a deployed network (e.g., Converge project)
 - This requires tight interconnection between radio processing infrastructure and the GPU farm but can be used to perform experiments not possible on the real network (large number of terminals). Can run on current cloud infrastructure (e.g. AWS).
 - Novel aspect, joint radio and digital twin. This requires proximity of HPC and real radio infrastructure.
- Example 2: Code analysis and bug fixing
 - Protocol implementations are bug-ridden. In the CI/CD Type 3 experiment, developers of OSS networking software can make use of SLICES-RI GPU farms for code analysis and bug fixing.
 - Today in projects like OAI, CI makes use of “simple” tools like cppcheck to analyze community contributions. Use of AI/ML tools will take this to another level.

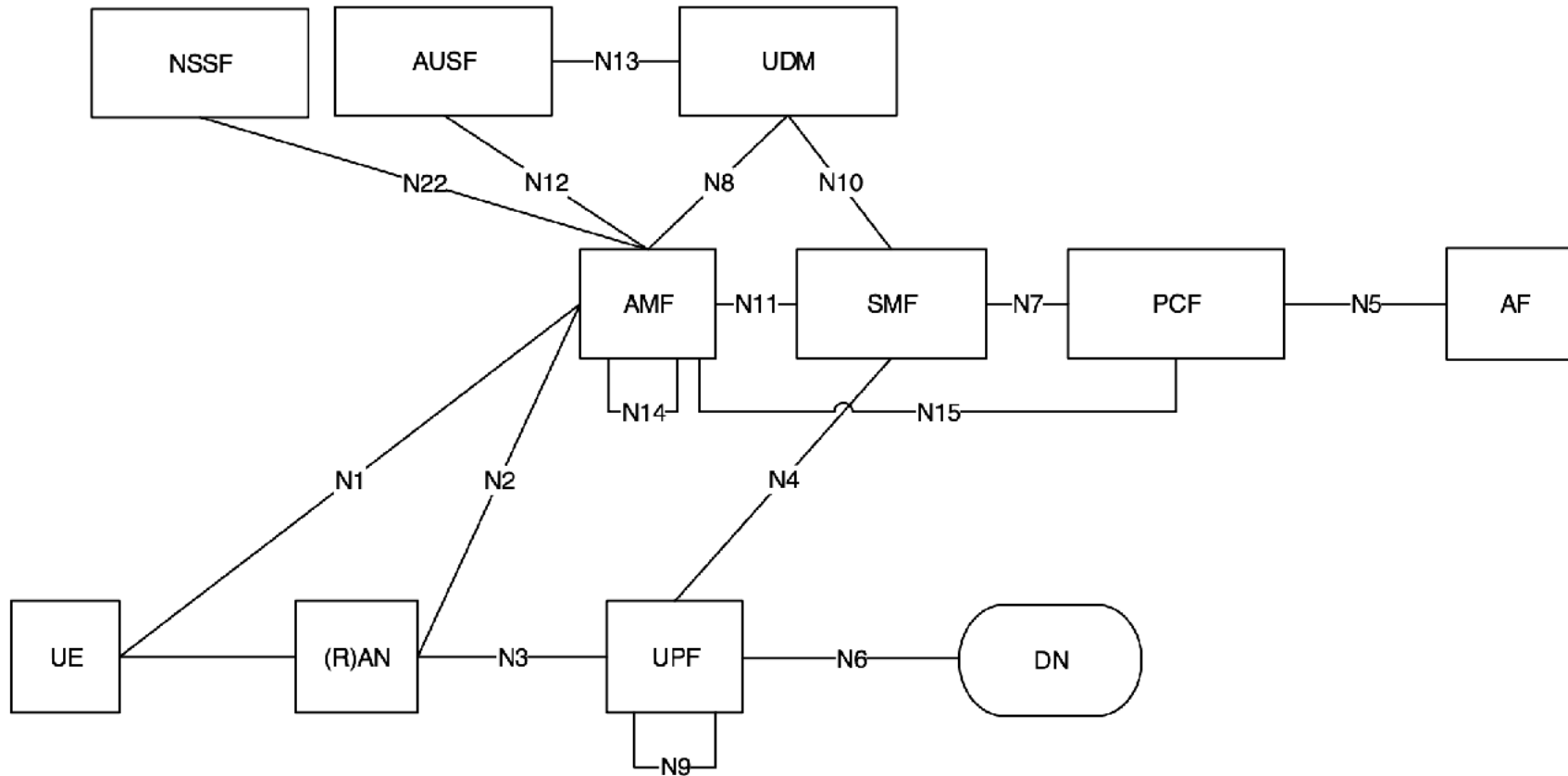
Why 5G?



5G key principles [1]

- Separate the User Plane functions from the Control Plane functions
- Modularize the function design
- Define the set of interactions between network functions as services
- Enable direct interactions between network functions
- Minimize dependencies between the Access Network and the Core Network
- Support "stateless" NFs (i.e., compute decoupled from storage)
- Support capability exposure

General core architecture



AF: Application Function
AMF: Access and Mobility Management Function
AUSF: Authentication Server Function
DN: Data Network (DN)
NSSF: Network Slice Selection Function
PCF: Policy Control Function
(R)AN: (Radio) Access Network
SMF: Session Management Function
UDM: Unified Data Management
UE: User Equipment
UPF: User Plane Function

Figure 4.2.3-2: Non-Roaming 5G System Architecture in reference point representation [1]

Split-RAN

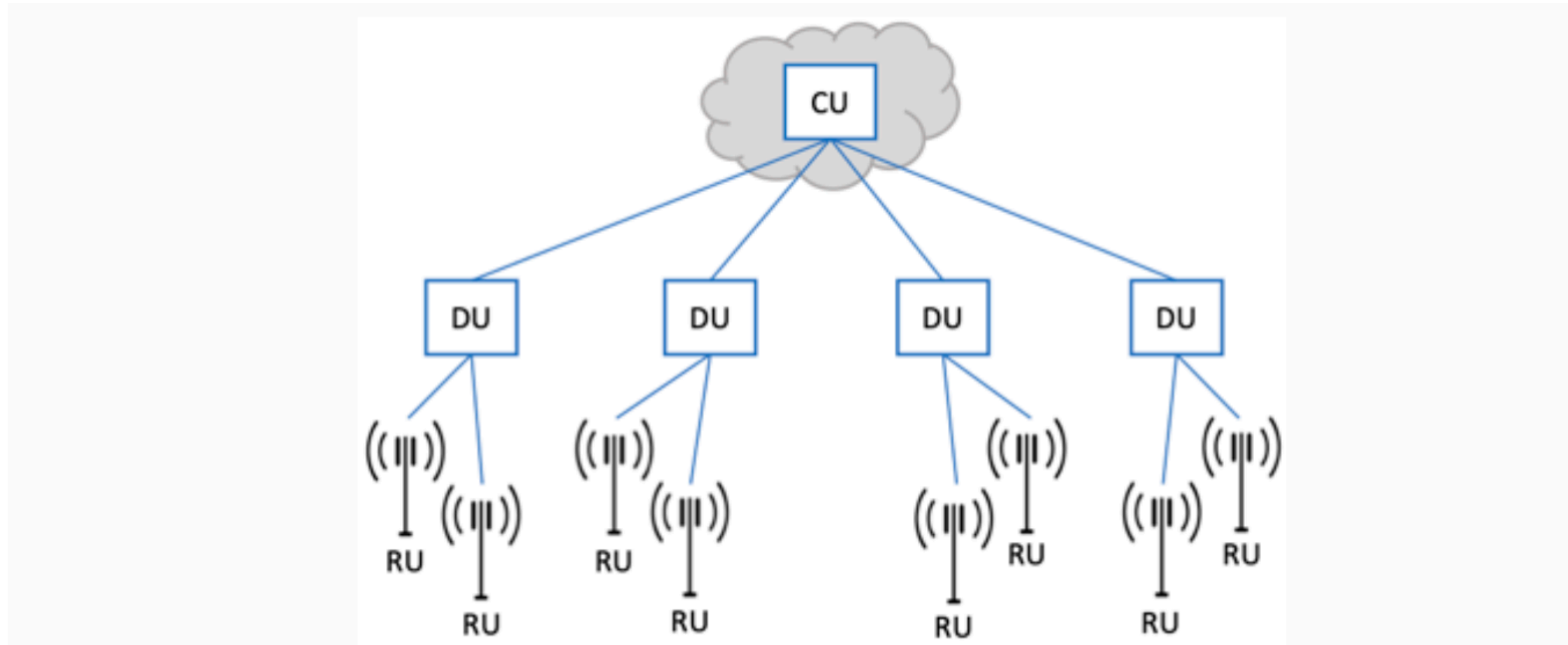


Figure 26. Split RAN hierarchy, with one CU serving multiple DUs, each of which serves multiple RUs. [2]

Split-RAN

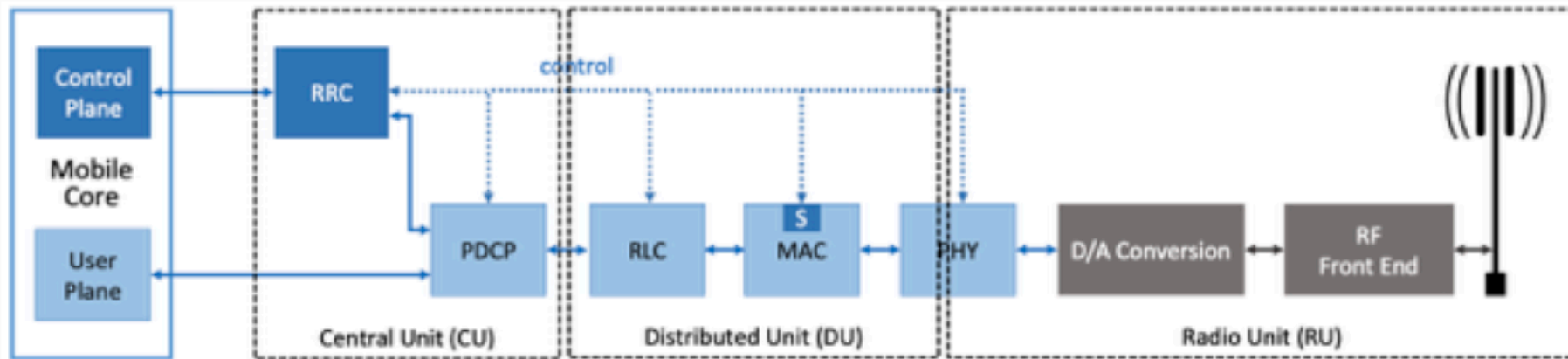
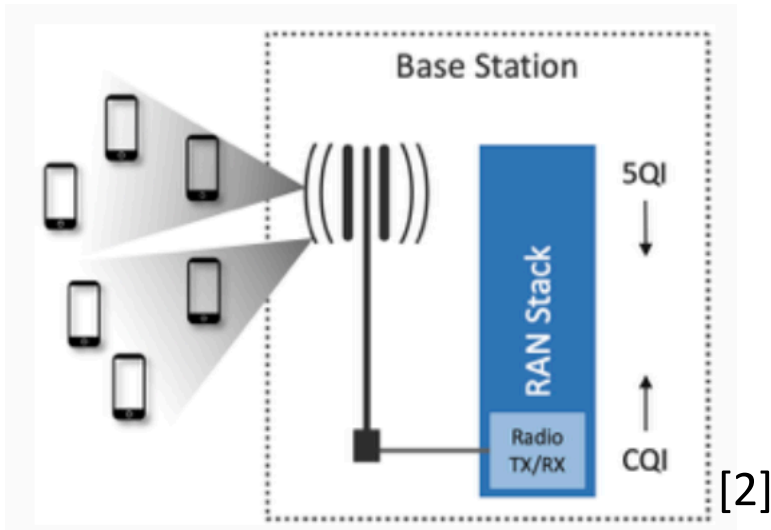


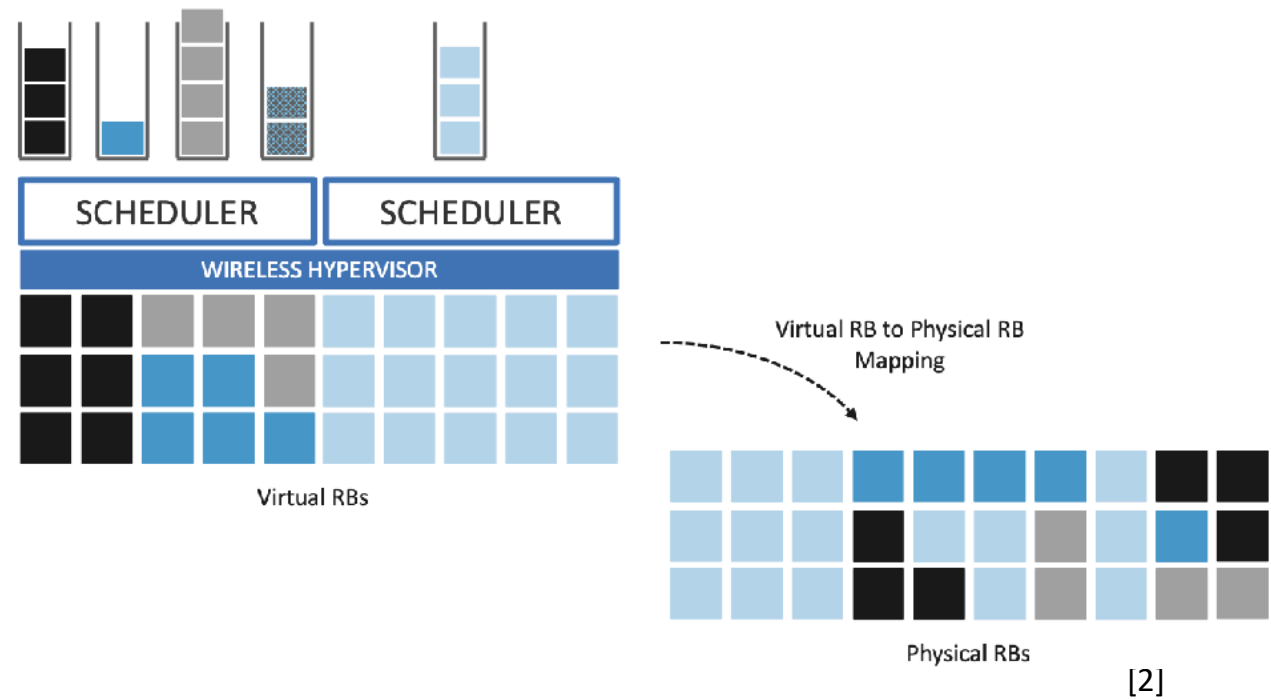
Figure 25. Split RAN processing pipeline distributed across a Central Unit (CU), Distributed Unit (DU), and Radio Unit (RU). [2]

Radio optimization

- High utilisation of spectrum, sub-millisecond control loop



5QI: 5G QoS Identifier
 CQI: Channel Quality Indicator



HW-based UPF implementations

- Intel Tofino silicon is discontinued
- DPDK, DDP, QAT... is not enough (~10Gbps, ~1ms)
- DPU/IPU-based solutions (e.g., Marvel OCTEON, Nvidia Bluefield 3) will fill this need
 - P4 may still be relevant but more conventional solutions (C/C++ control SW on ARM with HW accelerators)
- Now studying the use of such processors for UPF and/or CU-UP

HW-based UPF implementations

- Intel Tofino silicon is discontinued
- DPDK, DDP, QAT... is not enough (~10Gbps, ~1ms)

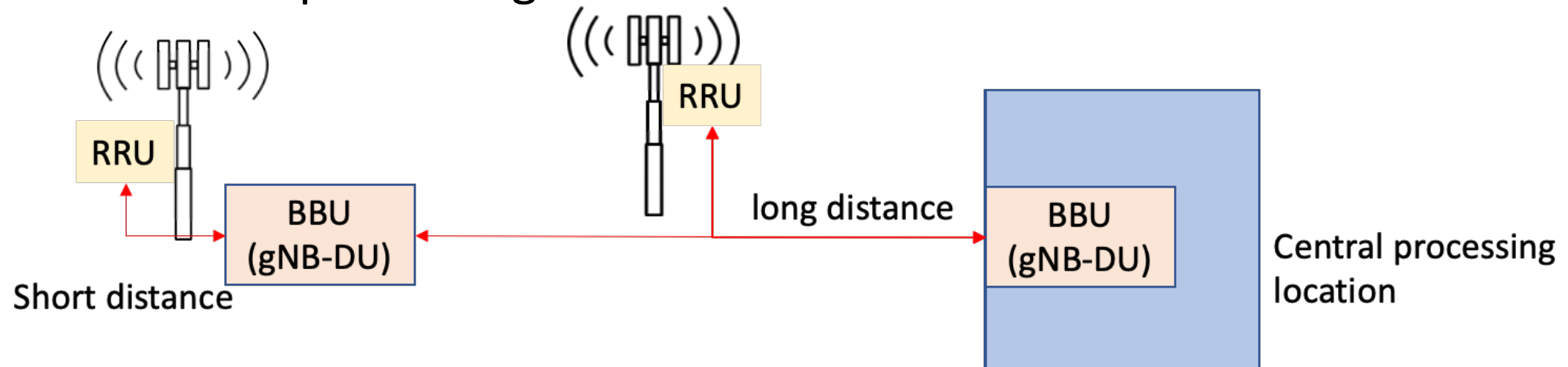
How to be cloud-nativize?

control SW on ARM with HW accelerators)

- Now studying the use of such processors for UPF and/or CU-UP

Fronthaul

- Transport of radio signals over a network:
 - to allow for statistical multiplexing of multiple radio sites sharing common fiber links
 - to share processing between radio-units and baseband units

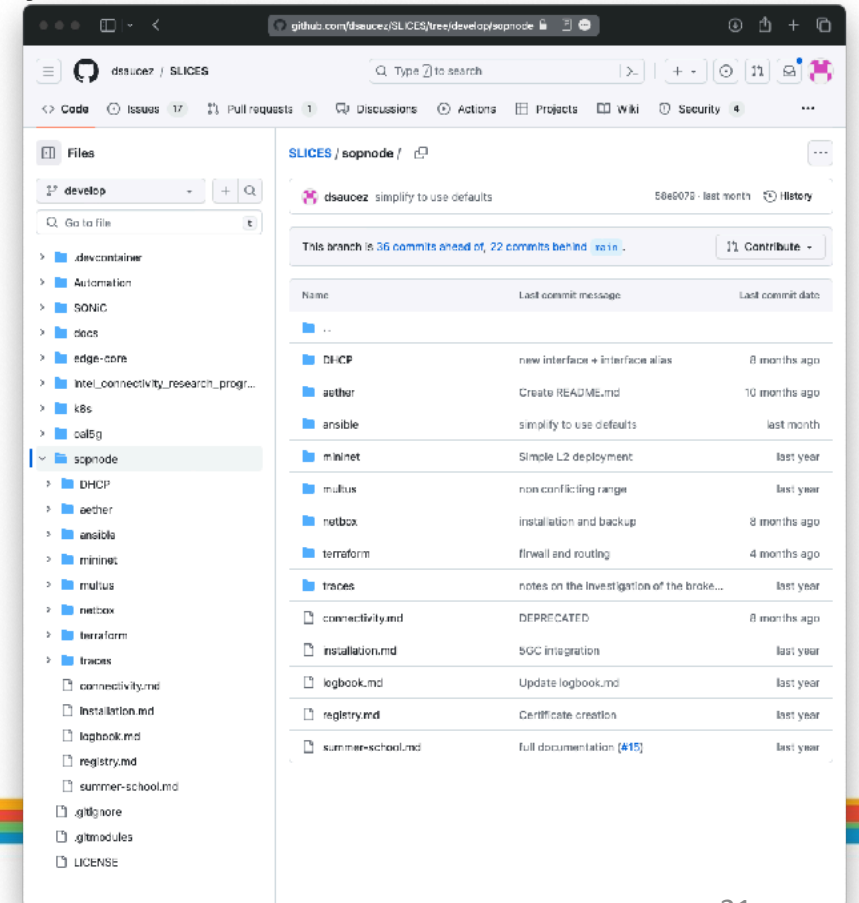


Fronthaul implementation

- O-RAN Open FHI in OpenAirInterface
- Library poptimized for Intel-based x86-64 (e.g., AVX512)
 - How to move to AMD and ARM?
- Complexity in network infrastructure
 - Raw Ethernet
 - Usually 3 VLAN (VMDq not enough, need sr-iov): management, control/user plane
 - PTP distribution and PTP HW time stamping on NICs

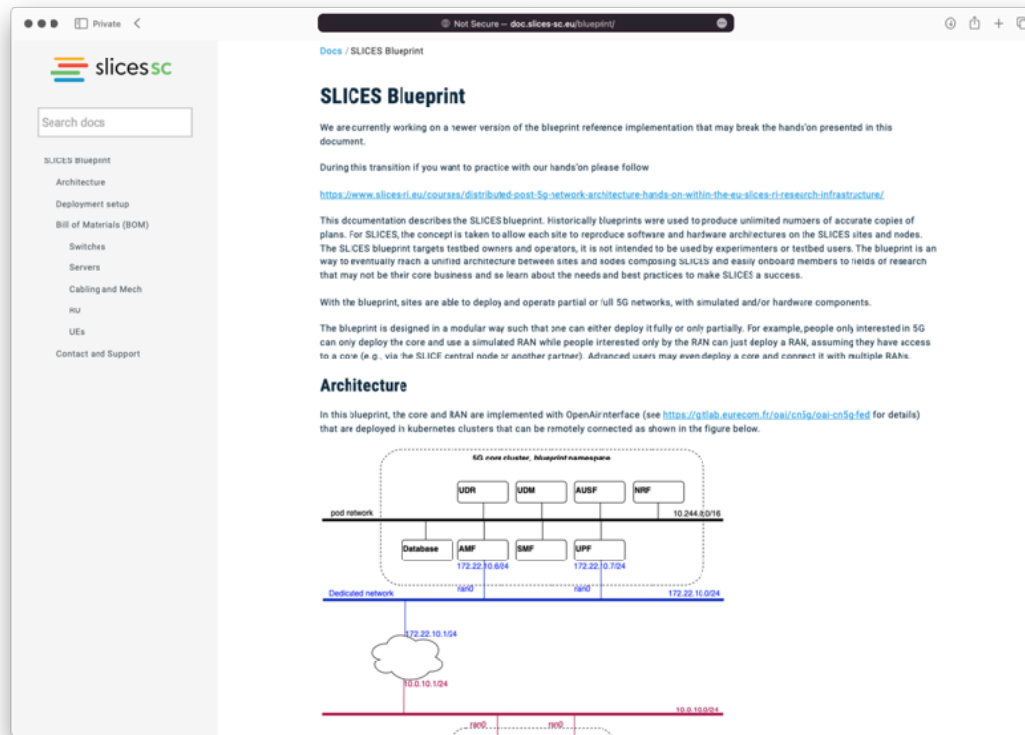
Reference implementation (1/3)

- Terraform (support of GCP) + Ansible for deployments
- kubernetes + docker to manage resources
- OpenAirInterface
 - 5G core implementation
 - RAN implementation
- Software and/or hardware
 - USRP/AW2S
 - Tofino/Tofino2 with SD-Fabric

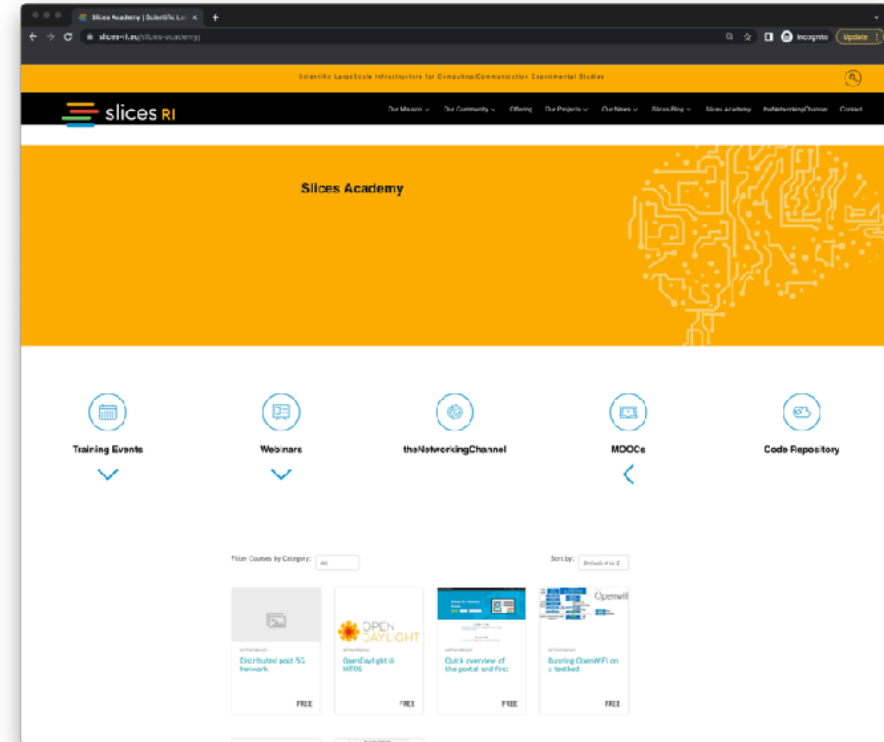


Reference implementation (2/3)

- Complete documentation and SLICES academy integration



The screenshot shows a web browser displaying the 'SLICES Blueprint' documentation. The page title is 'SLICES Blueprint' and it includes a search bar and a navigation menu on the left. The main content area contains text explaining the blueprint's purpose and a network architecture diagram. The diagram illustrates a multi-tier network structure with a 'pod network' at the top, a 'Dedicated network' in the middle, and a 'Public network' at the bottom. The pod network contains components like UDR, UDM, AUSF, and NRF. The dedicated network contains Database, AMF, SMF, and UPF. The public network is connected to the dedicated network via gateways.



The screenshot shows the 'SLICES Academy' website. The header features the 'slices RI' logo and navigation links. The main content area is titled 'Slices Academy' and features a large orange graphic of a brain with circuitry. Below this, there are five icons representing different resources: Training Events, Webinars, theNetworkingChannel, MOOCs, and Code Repository. At the bottom, there is a section for 'Filter Courses by Category' and 'Sort by', followed by a grid of course cards. The course cards include titles like 'Distributed post-5G Network', 'OpenSAILLIGHT: OpenSAILLIGHT @ 5G/6G', 'Quick overview of the post-5G and 6G', and 'Running OpenAPI on a testbed'. Each card is labeled 'FREE'.

Reference implementation (3/3)

- Continuous testing

Reference implementation (3/3)

• Continue

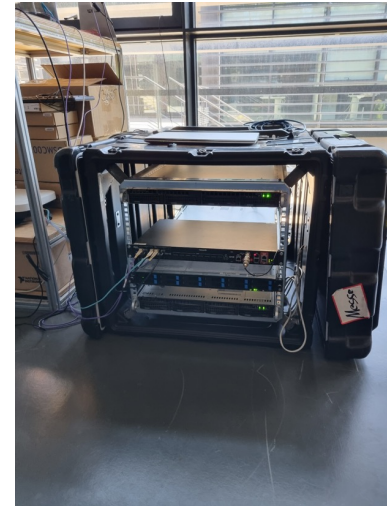
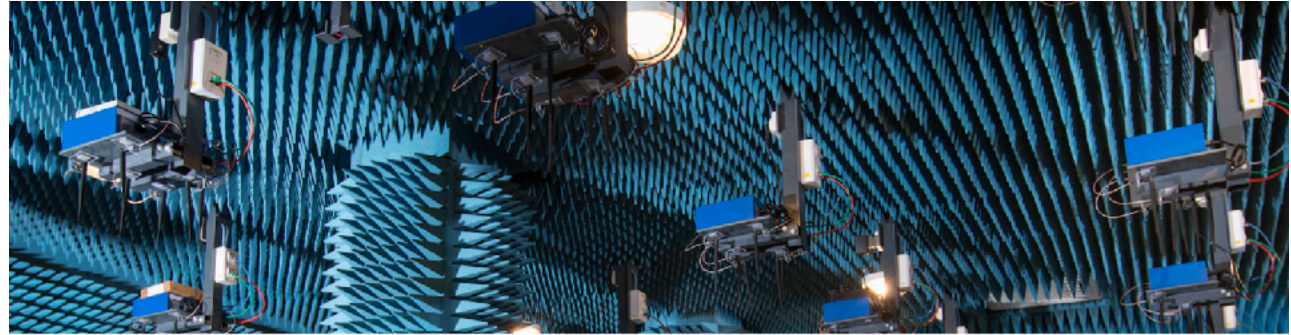
The screenshot displays the Azure DevOps web interface for a pipeline named 'dsaucez.SLICES'. The left sidebar shows the navigation menu with 'Pipelines' selected. The main content area shows a list of pipeline runs, all of which are successful (indicated by green checkmarks). Each run entry includes a description, a 'Stages' column with a green checkmark, and a completion time.

Description	Stages	Completion Time
#20231128.1 • Develop (#46) Scheduled for main @ 25f1839a	✓	17h ago 27m 49s
#20231127.1 • Develop (#46) Scheduled for main @ 25f1839a	✓	Yesterday 27m 36s
#20231126.1 • Develop (#46) Scheduled for main @ 25f1839a	✓	Sunday 30m 25s
#20231125.1 • Develop (#46) Scheduled for main @ 25f1839a	✓	Saturday 27m 39s
#20231124.1 • Develop (#46) Scheduled for main @ 25f1839a	✓	Friday 27m 28s
#20231123.1 • Develop (#46) Scheduled for main @ 25f1839a	✓	Thursday 27m 47s
#20231122.1 • Develop (#46) Scheduled for main @ 25f1839a	✓	Wednesday 27m 27s
#20231121.1 • Develop (#46) Scheduled for main @ 25f1839a	✓	21 Nov 27m 7s
#20231120.1 • Develop (#46) Scheduled for main @ 25f1839a	✓	20 Nov 27m 36s



slices

Lessons learned



Fronthaul

Diversity of deployments (1/2)

- On-prem
 - Core + HW RAN @ Eurecom
 - Core + HW RAN @ Inria Sophia
 - Core/RAN split
 - Local 200Gbps fiber: Core @ Inria Sophia / HW RAN @ Eurecom
 - International dedicated VLAN 1Gbps/10ms: Core @ imec / RAN @ Inria Sophia
 - Commercial Internet:
 - Core @ Sorbonne Université / RAN @ UTH
 - Core @ Sorbonne Université / RAN @ Inria Sophia
 - Split 7.2 Core + RAN @ Inria Sophia
- Public cloud aided
 - Core by Inria @ GCP / HW RAN @ Eurecom
 - Core by Inria @ GCP / HW RAN @ UTH
 - Core @ CNR / RAN by Inria @ GCP
- In testbeds
 - Summer school in TUM testbed, Virtual Wall



Diversity of deployments (2/2)

- Network interconnect
 - Commercial Internet accessible
 - Dedicated links
 - VPN service
- Identification / Authentication
 - Users based on the light federation from SLICES (OIDC) is ok
 - PKIs... how to unify certificates
- People test first in virtual environments (must support VM)

Doomed by dependencies

- Blueprint reference implementation relies on independent open source projects
 - all with their own agenda
 - dependencies brake frequently
 - freeze versions whenever possible
 - document dependency graph
 - test, test, test



Doomed by dependencies

- Bluepri source

- all w

- dep

- f

- c

- t

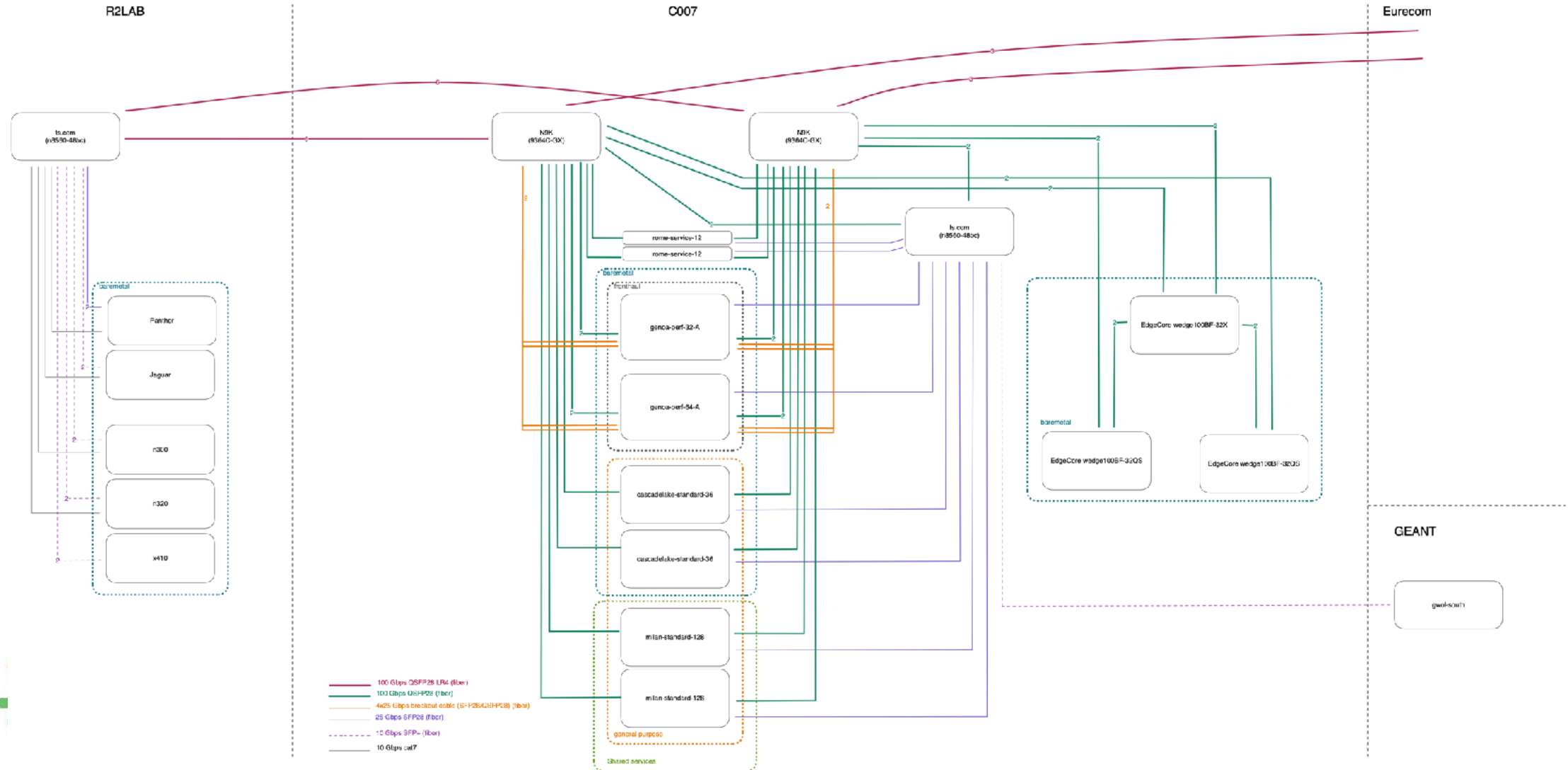


slices

The screenshot shows the Azure DevOps web interface. The breadcrumb navigation is 'anassdahchour / SLICESBlueprint / Pipelines / dsaucez.SLICES'. The left sidebar shows the 'Pipelines' menu item selected. The main content area displays the 'dsaucez.SLICES' pipeline with a 'Runs' tab active. The runs table shows the following data:

Description	Stages	Time
#20231128.1 • Develop (#46) Scheduled for main 25f1839a	✓	17h ago 27m 49s
#20231127.1 • Develop (#46) Scheduled for main 25f1839a	✓	Yesterday 27m 36s
#20231126.1 • Develop (#46) Scheduled for main 25f1839a	✓	Sunday 30m 25s
#20231125.1 • Develop (#46) Scheduled for main 25f1839a	✓	Saturday 27m 39s
#20231124.1 • Develop (#46) Scheduled for main 25f1839a	✓	Friday 27m 28s
#20231123.1 • Develop (#46) Scheduled for main 25f1839a	✓	Thursday 27m 47s
#20231122.1 • Develop (#46) Scheduled for main 25f1839a	✓	Wednesday 27m 27s
#20231121.1 • Develop (#46) Scheduled for main 25f1839a	✓	21 Nov 27m 7s
#20231120.1 • Develop (#46) Scheduled for main 25f1839a	✓	20 Nov 27m 36s

Heterogenous hw collection



Source of truth with netbox...

The screenshot displays the NetBox interface for a specific rack. The left sidebar contains navigation menus for Organization, Sites, RACKS, TENANCY, CONTACTS, Connections, Wireless, IPAM, Overlay, Virtualization, Circuits, Power, and Other. The main content area shows the following details:

- Rack rack-c007**: Created 2023-01-04 17:06, Updated 8 months, 1 week ago.
- Metadata Table:**

Site	fr / saphanode-imia
Location	c007
Facility ID	—
Tenant	—
Status	active
Role	—
Description	—
Serial Number	—
Asset Tag	—
Devices	12
Space Utilization	28.6%
Power Utilization	0.0%
- Dimensions Table:**

Type	—
Width	19 Inches
Height	42U (ascending)
Outer Width	—
Outer Depth	—
Mounting Depth	—
Rack Weight	—
Maximum Weight	—
Total Weight	0 Kilograms (0 Pounds)
- Rack Diagrams:** Two diagrams, 'Front' and 'Rear', show a 42U rack with devices placed at various heights. The 'Front' diagram shows devices like saphnode-w1, saphnode-w2, saphnode-w3, saphnode-sw1, saphnode-sw2, saphnode-sw3, saphnode-pdu-bas, and saphnode-pdu-bes. The 'Rear' diagram shows devices like saphnode-w1, saphnode-w2, saphnode-w3, saphnode-sw1, saphnode-sw2, saphnode-sw3, saphnode-pdu-bas, and saphnode-pdu-bes. Some devices are highlighted in red with diagonal stripes, indicating they are not yet fully configured or are in a specific state.

Source of truth with netbox...

The image displays two overlapping screenshots of the NetBox web interface. The background screenshot shows the 'Rack rack-c007' view, which includes a sidebar with navigation options like Organization, Sites, Racks, and Connections. The main content area shows details for the rack, including its location (c007), status (Active), and dimensions. The foreground screenshot shows the 'Devices' view for the same rack, displaying a table of devices with columns for Name, Status, Tenant, Site, Location, Rack, Role, Manufacturer, Type, and IP Address. The table lists various devices such as jgaur, n300, n320, panther, patch-panel-c007, patch-panel-r2lab, sspnode-r1, sspnode-pdu-bas, sspnode-pdu-haut, sspnode-sw1, sspnode-sw2, sspnode-sw3, sspnode-w1, sspnode-w2, sspnode-w3, sspnode-z1, switch-mgmt, and switch-rd1e.

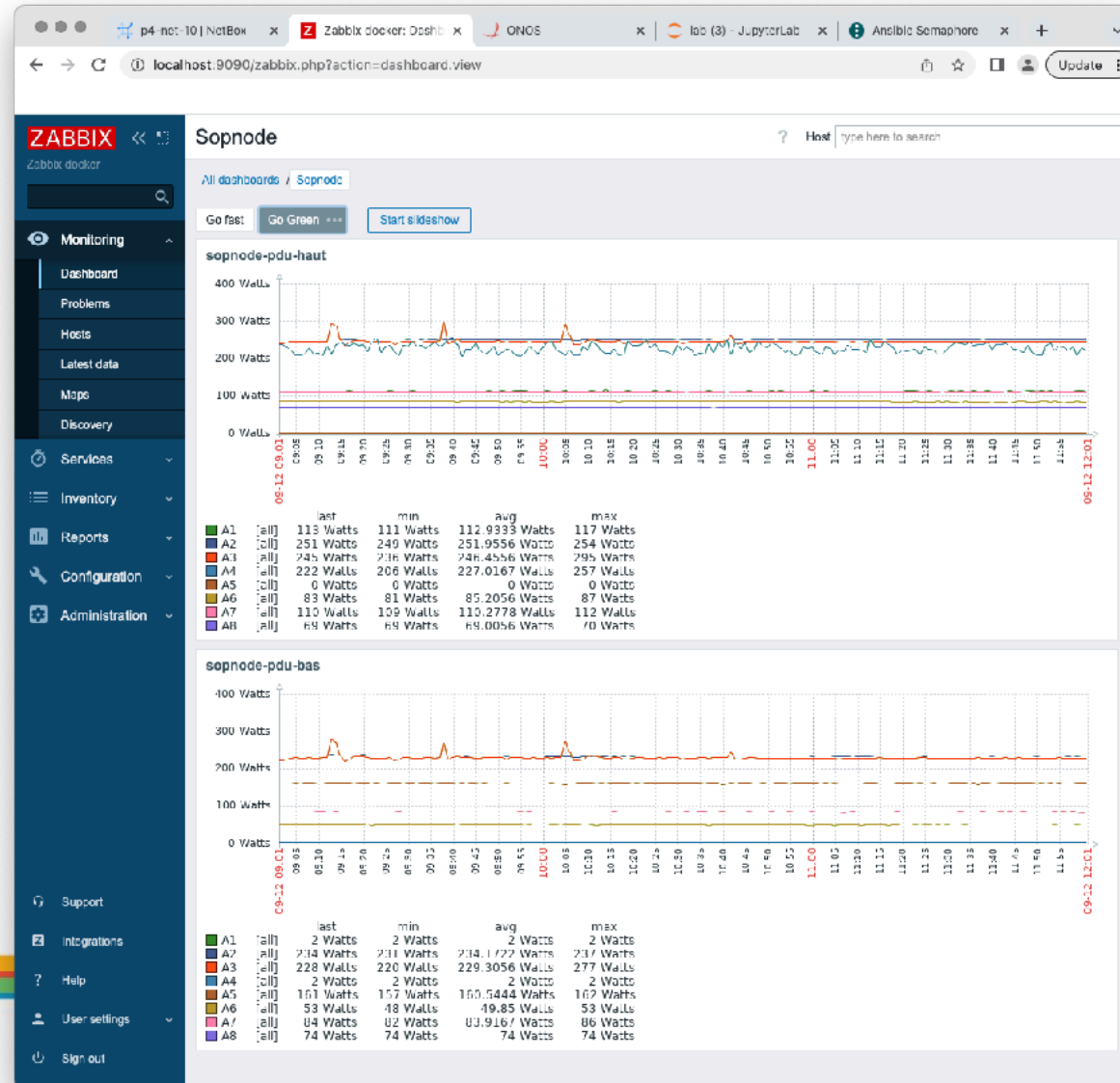
Name	Status	Tenant	Site	Location	Rack	Role	Manufacturer	Type	IP Address
jgaur	Active	---	sophianode-aria	r2lab	rack-r2lab	RRU	AWZS	jgaur	---
n300	Active	---	sophianode-aria	r2lab	rack-r2lab	USRP	Ettus	n300	---
n320	Active	---	sophianode-aria	r2lab	rack-r2lab	USRP	Ettus	n320	---
panther	Active	---	sophianode-aria	r2lab	rack-r2lab	RRU	AWZS	panther	---
patch-panel-c007	Active	---	sophianode-aria	c007	rack-c007	patch-panel	no-name	patch_panel	---
patch-panel-r2lab	Active	---	sophianode-aria	r2lab	rack-r2lab	patch-panel	no-name	patch_panel	---
sspnode-r1	Active	---	sophianode-aria	c007	rack-c007	compute-node	Dell	PowerEdge R640	---
sspnode-pdu-bas	Active	---	sophianode-aria	c007	rack-c007	PDU	EATON	EMA1-2B	138.96.245.30/24
sspnode-pdu-haut	Active	---	sophianode-aria	c007	rack-c007	PDU	EATON	EMA1-2B	138.96.245.31/24
sspnode-sw1	Active	---	sophianode-aria	c007	rack-c007	programmable-switch	Edgcore	Wedge 1009F-32QS	138.96.245.12/24
sspnode-sw2	Active	---	sophianode-aria	c007	rack-c007	programmable-switch	Edgcore	Wedge 1009F-32QS	138.96.245.12/24
sspnode-sw3	Active	---	sophianode-aria	c007	rack-c007	programmable-switch	Edgcore	Wedge 1009F-32QS	138.96.245.12/24
sspnode-w1	Active	---	sophianode-aria	c007	rack-c007	compute-node	Dell	PowerEdge R640	---
sspnode-w2	Active	---	sophianode-aria	c007	rack-c007	compute-node	Dell	PowerEdge R640	---
sspnode-w3	Active	---	sophianode-aria	c007	rack-c007	compute-node	Dell	PowerEdge R640	---
sspnode-z1	Active	---	sophianode-aria	c007	rack-c007	compute-node	Dell	PowerEdge R630	---
switch-mgmt	Active	---	sophianode-aria	c007	rack-c007	IS-switch	FS.com	n6560-40bc	---
switch-rd1e	Active	---	sophianode-aria	r2lab	rack-r2lab	IS-switch	FS.com	n6560-40bc	---

Source of truth with netbox...

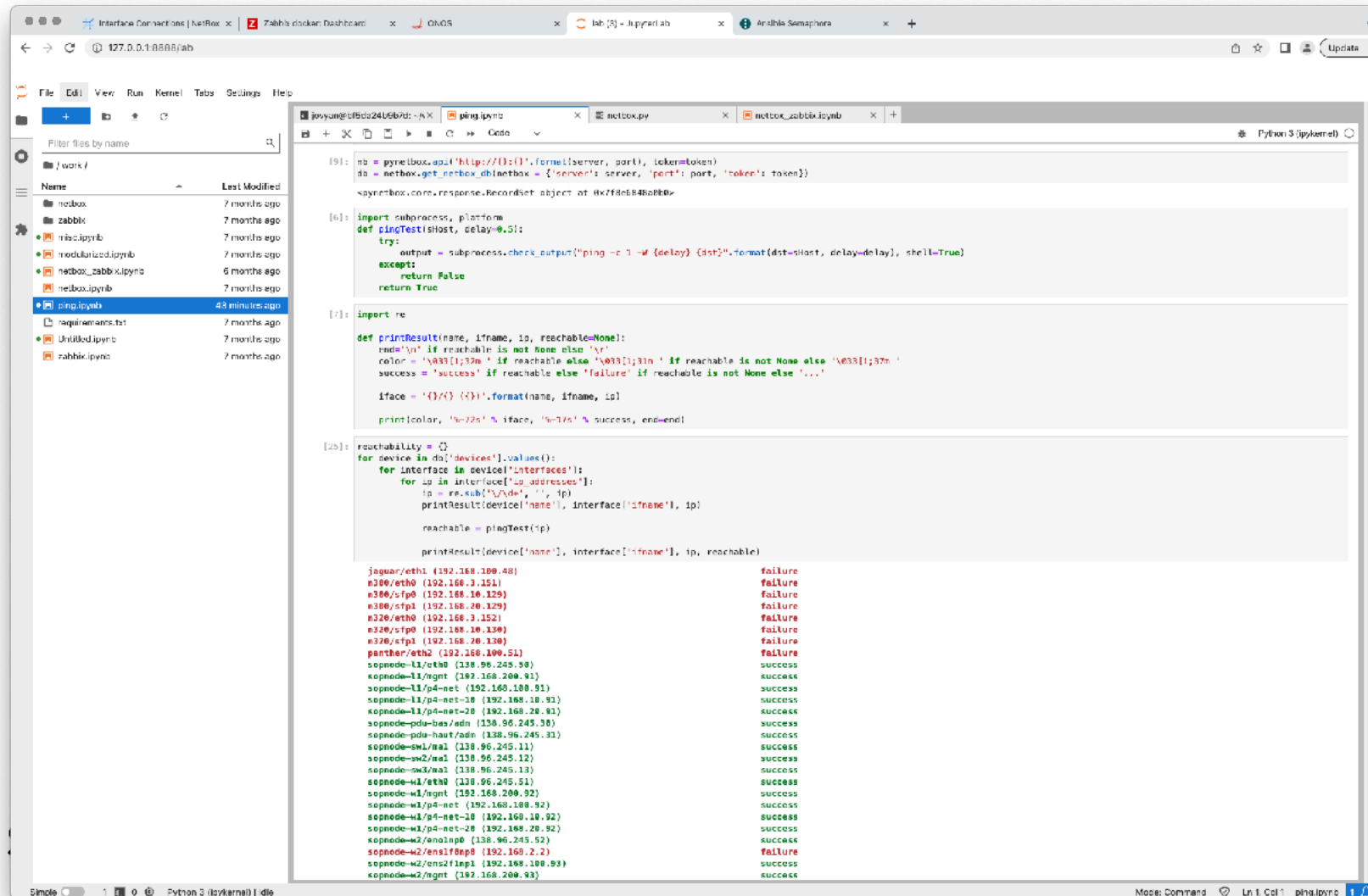
The image displays three overlapping screenshots of the NetBox web interface, illustrating its role as a source of truth for network configuration. The background screenshot shows the 'Rack rack-c007' view, detailing site information, rack specifications, and dimensions. The middle screenshot shows the 'Devices' view, listing various network devices like 'lqgaur', 'n300', and 'sopnode-11' with their status and active components. The foreground screenshot shows the 'Interface Connections' view, providing a detailed table of connections between devices and their interfaces.

Device	Name	Connection	Reachable
lqgaur	eth1	switch-radio > TFGigabitEthernet 0/15	✓
n300	sfp0	switch-radio > TFGigabitEthernet 0/38	✓
n300	sfp1	switch-radio > TFGigabitEthernet 0/37	✓
n520	sfp0	switch-radio > TFGigabitEthernet 0/25	✓
n520	sfp1	switch-radio > TFGigabitEthernet 0/26	✓
panther	eth0	switch-radio > TFGigabitEthernet 0/5	✓
panther	eth2	switch-radio > TFGigabitEthernet 0/5	✓
sopnode-11	mgmt	switch-mgmt > HundredGigabitEthernet 0/49-1	✓
sopnode-11	p4-net	sopnode-sw2 > 103	✓
sopnode-11	p4-net-10	sopnode-sw2 > 100	✓
sopnode-11	p4-net-20	sopnode-sw2 > 101	✓
sopnode-sw1	14/0	switch-radio > HundredGigabitEthernet 0/49	✓
sopnode-sw1	28/0	sopnode-sw2 > 29/0	✓
sopnode-sw1	100	sopnode-w1 > p4-net-10	✓
sopnode-sw1	101	sopnode-w1 > p4-net-20	✓
sopnode-sw1	103	sopnode-w1 > p4-net	✓
sopnode-sw1	201	sopnode-w2 > p4-net-20	✓
sopnode-sw1	202	sopnode-w2 > rns2f1np1	✓
sopnode-sw1	301	sopnode-w3 > rns3f1np1	✓
sopnode-sw1	302	sopnode-w3 > p4-net-20	✓
sopnode-sw2	25/0	sopnode-sw3 > 25/0	✓
sopnode-sw2	28/0	sopnode-sw1 > 29/0	✓

... to automate monitoring



... to run experiments in a programatic way



```
[19]: nb = pyNetBox.api('https://10.10.10.10', format='server', port=80, token='token')
      db = netbox.get_netbox_db(netbox={'server': 'server', 'port': port, 'token': 'token'})
      <pyNetBox.core.response.RecordSet object at 0x7f8e648a0800>

[6]: import subprocess, platform
      def pingTest(dstHost, delay=0.5):
          try:
              output = subprocess.check_output("ping -c 1 -W {delay} {dst}" .format(dst=dstHost, delay=delay), shell=True)
          except:
              return False
          return True

[7]: import re

      def printResult(name, ifname, ip, reachable=None):
          color = '\n' if reachable is not None else '\n'
          color = '\033[1;32m' if reachable else '\033[1;31m' if reachable is not None else '\033[1;37m'
          success = 'success' if reachable else 'failure' if reachable is not None else ''

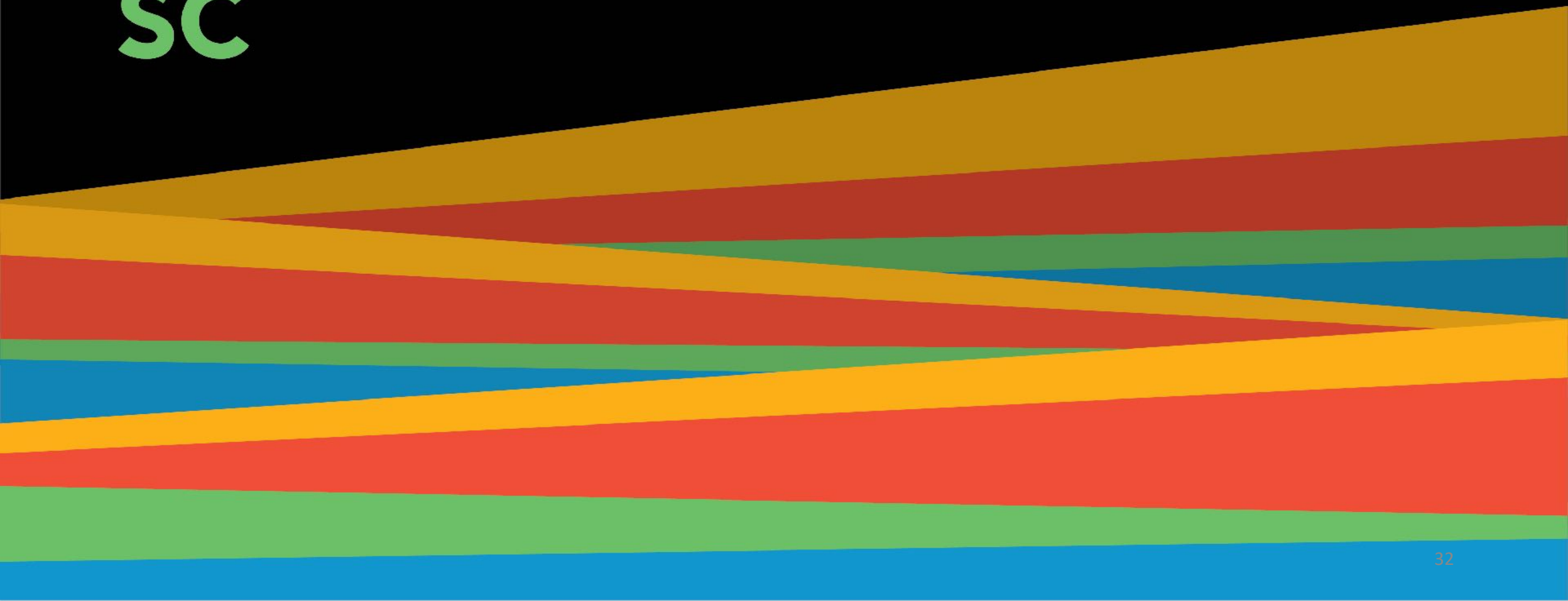
          iface = '{}/{}'.format(name, ifname, ip)
          print(color, '%-25s' % iface, '%-15s' % success, end='\n')

[25]: reachability = {}
      for device in db['devices'].values():
          for interface in device['interfaces']:
              for ip in interface['ip_addresses']:
                  ip = re.sub('\s+', '', ip)
                  printResult(device['name'], interface['ifname'], ip)

                  reachable = pingTest(ip)
                  printResult(device['name'], interface['ifname'], ip, reachable)

      jaguar/eth1 (192.168.100.48) failure
      n300/eth0 (192.168.3.151) failure
      n300/sfp0 (192.168.10.129) failure
      n300/sfp1 (192.168.20.129) failure
      n320/eth0 (192.168.3.152) failure
      n320/sfp0 (192.168.10.130) failure
      n320/sfp1 (192.168.20.130) failure
      panther/eth2 (192.160.100.51) failure
      s0pnode-11/eth0 (138.96.245.90) success
      s0pnode-11/ngnt (192.168.200.91) success
      s0pnode-11/p4-net (192.160.100.91) success
      s0pnode-11/p4-net-10 (192.168.10.91) success
      s0pnode-11/p4-net-20 (192.168.20.91) success
      s0pnode-pdu-bas/adm (138.96.245.30) success
      s0pnode-pdu-haut/adm (138.96.245.31) success
      s0pnode-sw1/ma1 (138.96.245.11) success
      s0pnode-sw2/ma1 (138.96.245.12) success
      s0pnode-sw3/ma1 (138.96.245.13) success
      s0pnode-w1/eth0 (138.96.245.51) success
      s0pnode-w1/ngnt (192.168.200.92) success
      s0pnode-w1/p4-net (192.168.100.92) success
      s0pnode-w1/p4-net-10 (192.160.10.92) success
      s0pnode-w1/p4-net-20 (192.168.20.92) success
      s0pnode-w2/eno1np0 (138.96.245.52) success
      s0pnode-w2/ens1f0np0 (192.160.2.2) failure
      s0pnode-w2/ens1f1np1 (192.168.100.93) success
      s0pnode-w2/ngnt (192.168.200.93) success
```

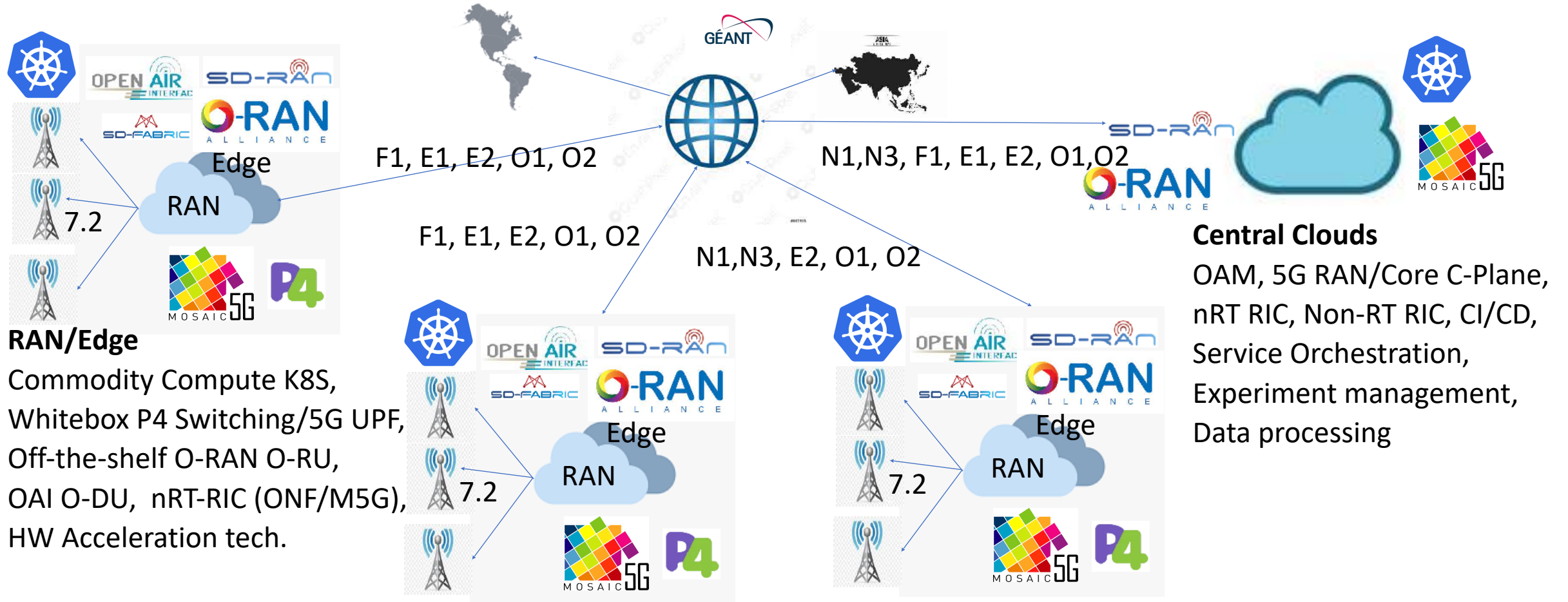




Backup



Current state



RAN/Edge
 Commodity Compute K8S,
 Whitebox P4 Switching/5G UPF,
 Off-the-shelf O-RAN O-RU,
 OAI O-DU, nRT-RIC (ONF/M5G),
 HW Acceleration tech.

Central Clouds
 OAM, 5G RAN/Core C-Plane,
 nRT RIC, Non-RT RIC, CI/CD,
 Service Orchestration,
 Experiment management,
 Data processing

post5G Experimentation in SLICES

- SLICES first 4-5 years: evolve beyond 5G using **open** 5G technologies on large-scale end-to-end platforms
- Focus on technologies targeting integration of disaggregated post5G RAN and core with cloud-native deployment framework
- Integration of new applications on experimental post5G infrastructure (SNS C/D)

User-Plane Function blueprints

- Software UPFs are readily available in OSS (OAI SPGW-u, BESS (with free5gcore), Traveling VPP...)
- P4HW UPFs with SD-Fabric

