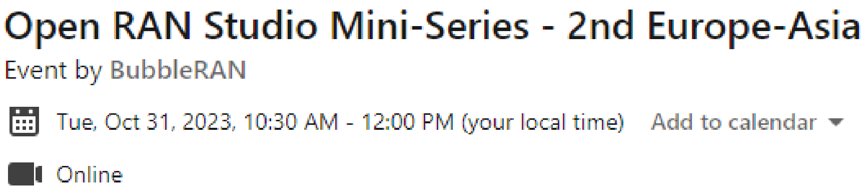


Open RAN Studio Mini-Series - 2nd Europe-Asia Edition

Event by BubbleRAN

Tue, Oct 31, 2023, 10:30 AM - 12:00 PM (your local time) Add to calendar ▾

Online

ORS Mini-Series

Episode 02: 5G Open RAN Observability at Scale
Provided by BubbleRAN

Tuesday, 31 October 2023

© BubbleRAN

Zoom Interaction Rules

1. Please add your affiliation to your zoom name, e.g. Navid Nikaein (BubbleRAN/Eurecom/OSA)
2. Please use Zoom Chat for any questions or comments outside of the Q&A session. This is highly recommended to be able to answer all the questions. The team will reply to you.
3. Please always mute the microphone and disable Video all the time to minimize background noise.
4. Please enable your video and audio when asking a question during the Q&A.
5. The recorded video will be published on the BubbleRAN Youtube channel. We will send a separate email with all the materials.

GDPR Notice: By joining this webinar, you are agreeing that your name and voice could appear in the recorded materials and published online without future consent.

© BubbleRAN

§

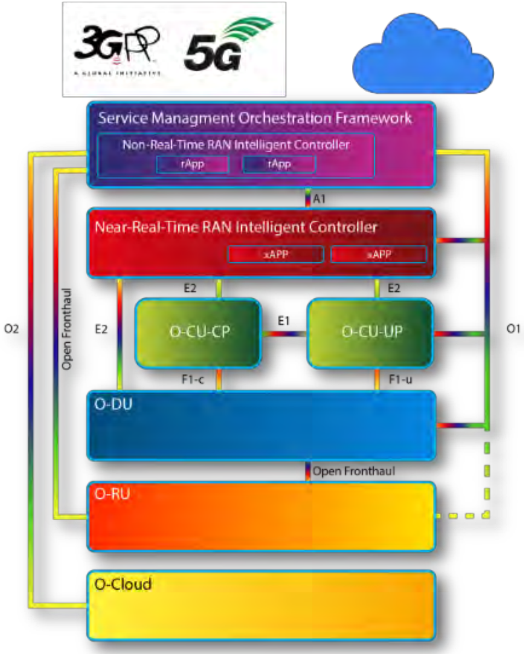
ORS Mini-Series

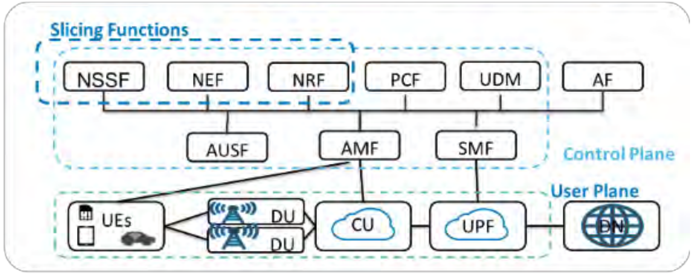
Importance, organization, and goals

Why Open RAN Studio Mini-Series?

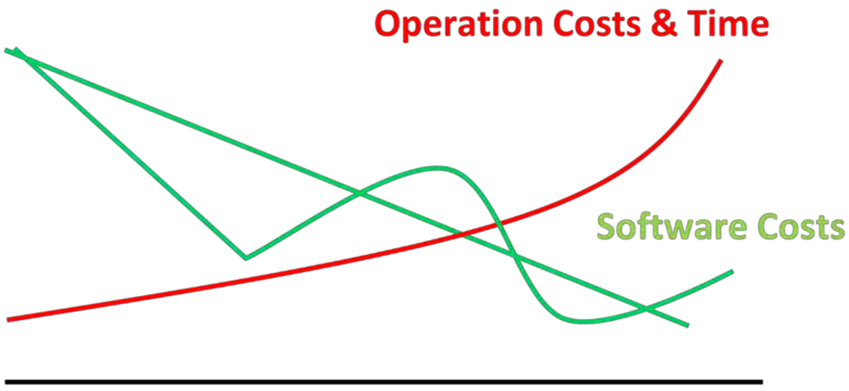
1. Share Knowledge
2. Identify New Challenges/Features
3. Accelerate the R&D lifecycle from Idea to PoC
4. Showcase Ideas and Validate use-cases
5. Foster Academia and Industry Collaboration

Terraform
Open Source MANO
NEPHIO
ONAP
OPEN NETWORK AUTOMATION PLATFORM
MOSAIC 5G
SRSRAN
OPEN AIR INTERFACE
CLOUD NATIVE
COMPUTING FOUNDATION
O-RAN
ALLIANCE
THE LINUX FOUNDATION

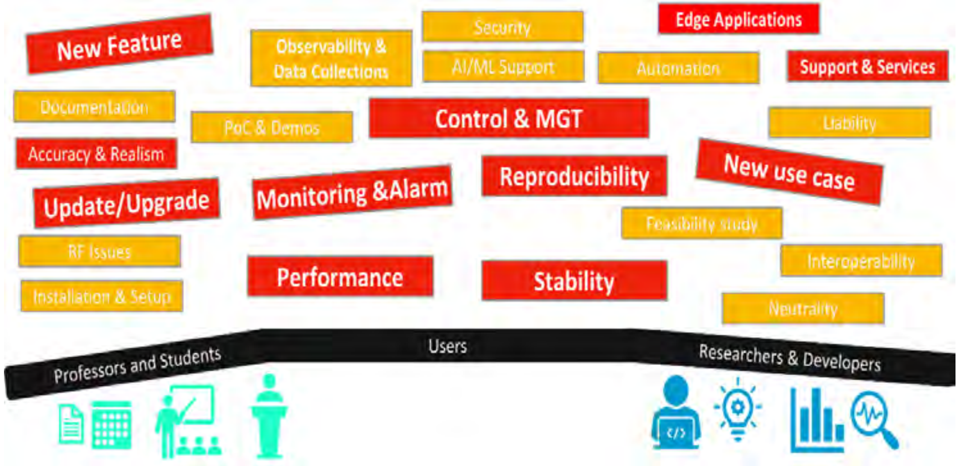

3GPP
A GLOBAL INITIATIVE
5G
Service Managment Orchestration Framework
Non-Real-Time RAN Intelligent Controller
rApp
rApp
A1
Near-Real-Time RAN Intelligent Controller
xAPP
xAPP
E2
E2
E2
O2
O1
Open Fronthaul
O-CU-CP
E1
O-CU-UP
F1-c
F1-u
O-DU
Open Fronthaul
O-RU
O-Cloud



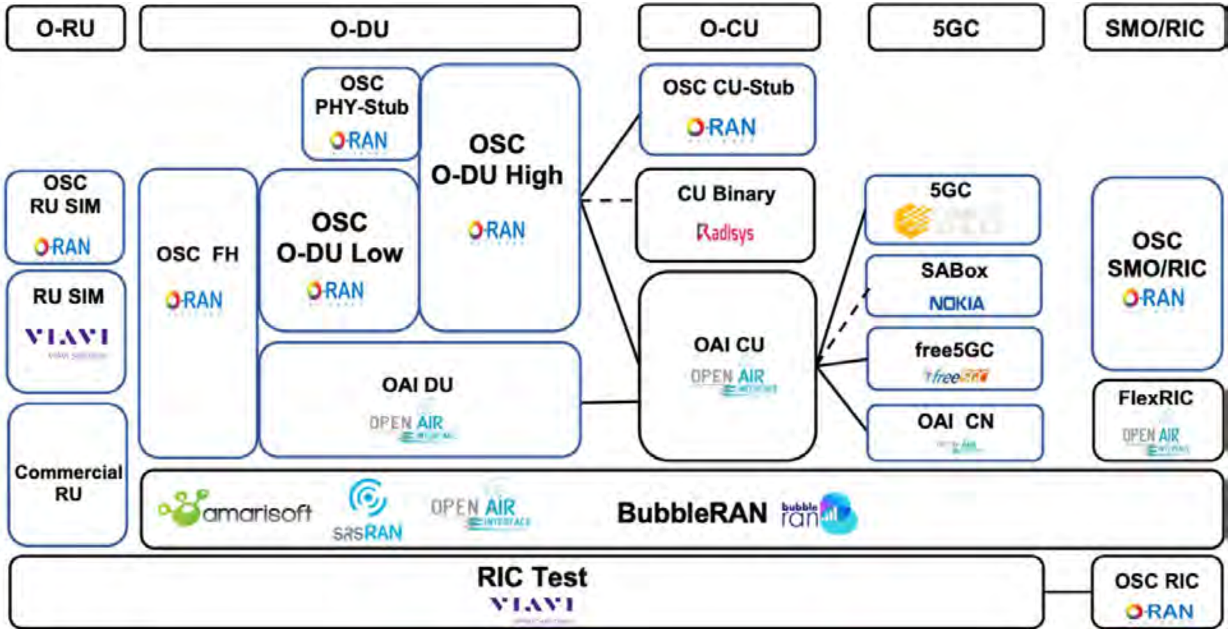
5G/6G Open Source Ecosystem are complex!

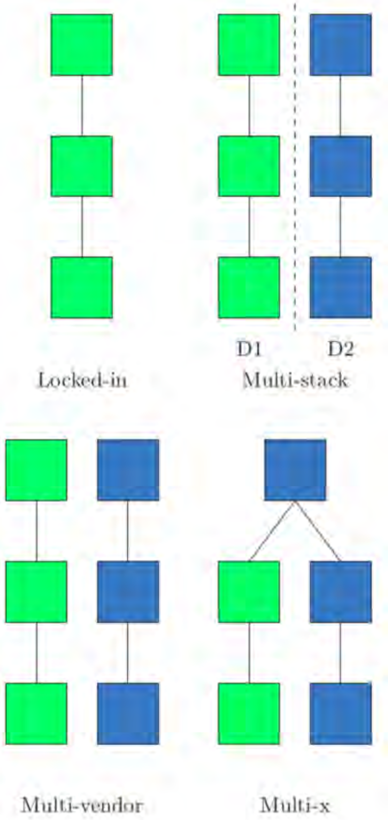


Open Source is not always a **positive sum gain**, for example O-RAN !

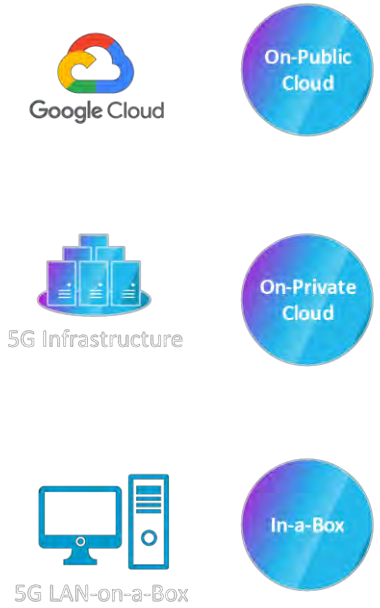


Open/Free Source Software are Becoming Expensive!





Example of a testbed at NTUST/NYCU (Taiwan)



Google Cloud

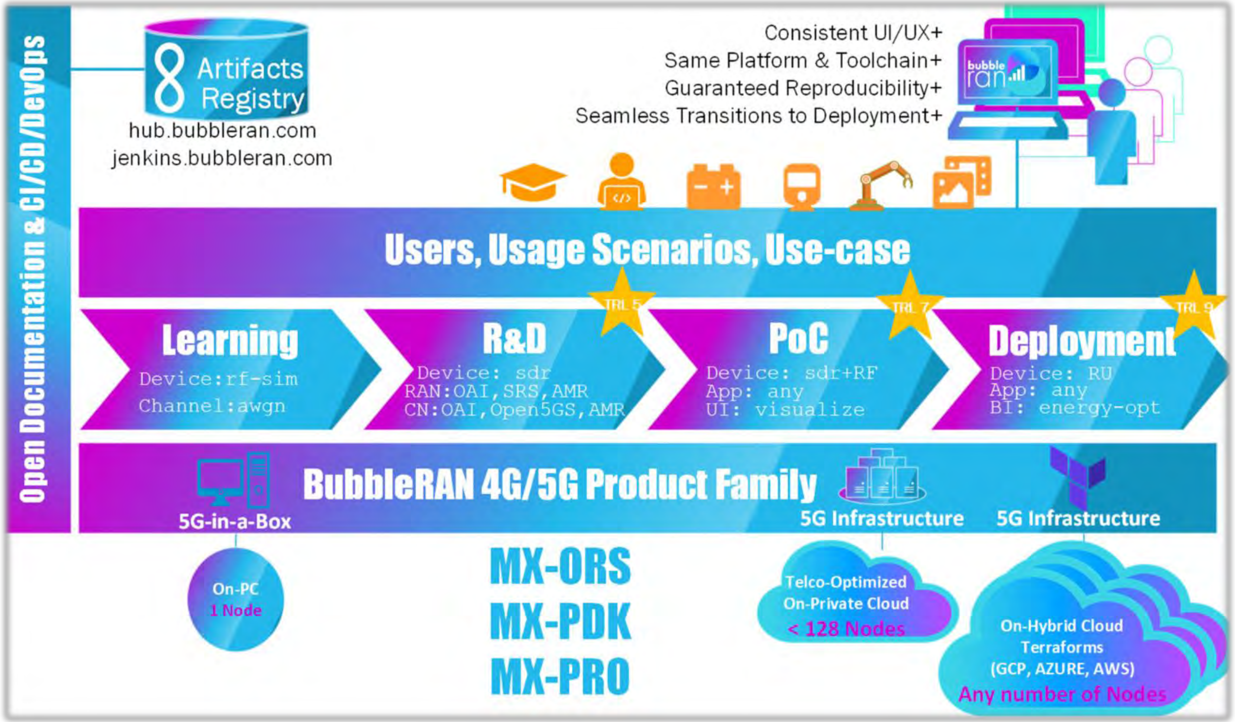
On-Public Cloud

5G Infrastructure

On-Private Cloud

5G LAN-on-a-Box

In-a-Box



Open Documentation & CI/CD/DevOps

Artifacts Registry

hub.bubbleran.com

jenkins.bubbleran.com

Consistent UI/UX+
Same Platform & Toolchain+
Guaranteed Reproducibility+
Seamless Transitions to Deployment+

Users, Usage Scenarios, Use-case

Learning
Device:rf-sim
Channel:awgn

R&D
TRL 5
Device: sdr
RAN:OAI,SRS,AMR
CN:OAI,Open5GS,AMR

PoC
TRL 7
Device: sdr+RF
App: any
UI: visualize

Deployment
TRL 9
Device: RU
App: any
BI: energy-opt

BubbleRAN 4G/5G Product Family

5G-in-a-Box

5G Infrastructure

5G Infrastructure

On-PC
1 Node

MX-ORS
MX-PDK
MX-PRO

Telco-Optimized
On-Private Cloud
< 128 Nodes

On-Hybrid Cloud
Terraforms
(GCP, AZURE, AWS)
Any number of Nodes

What is Open RAN Studio?

© BubbleRAN

Open RAN Studio Global Presence (2021-2023)

Episode title

Backgrounds & Resources

Guided Instructions

Basic + Bonus Questions

Report & Feedbacks

"Working with open RAN studio was interesting as it is easy to see the different elements of the open RAN and operate them through the CLI"

"I appreciated especially the first part about the traffic generation."

Regarding the labs, I think they helped in understanding the theoretical part. Personally, I have learned a lot from the class because I had very little experience in mobile networking."

"When we started the lab activity everything became much clearer for me and I was able to understand much better the communication and interaction between the different components in 4G and 5G communications."

"I've learnt through this class what are the requirements to connect an UE to the gNB and then the CN and can diagnose what could be the issue in case of failure."

"While the labs were challenging and time-consuming, they proved to be immensely valuable. They were consistent and highly beneficial in reinforcing the theoretical knowledge gained during the lectures. The labs provided a hands-on experience that significantly enhanced my practical skills and understanding of the tool."

Open RAN Studio Goals

1. Empower communities and organizations to accelerate the adoption of modern technologies
2. Solid ground for tutoring the next generation researchers and engineers
3. Reproducible/verifiable and consistent outcomes for teaching and research
4. Affordable and accessible means for education and research
5. Opening new possibilities and dimensions via multi-disciplinary research
    a. Cloud, Edge
    b. AI/ML, Data Science,
    c. Open RAN, 5G/6G
    d. Security
    e. Applications, Use-case, …

Today's Agenda and Speakers

Part 1 (45 minutes)

1. Open RAN Studio: Features and Bronze Release Notes
2. Non-RT RIC: Architecture and rApps call flow
3. OAM: How to design and deploy a 5G Open RAN network on GKE

Break (5 minutes)

Part 2 (45 minutes)

1. xApp lifecycle: RAN slicing use-cases
2. Data Analytics: Large-scale 5G Open RAN deployment
3. DevOps xApp: Interactive xApp
4. Observability: Data flow processing
5. Guest Demo (10 minutes)
   a. Interoperability between Open RAN Studio and OSC DU
6. Closing remarks and Q&A (10 minutes)

Alireza
BubbleRAN
Product Manager

Ilias
Eurecom
SMO expert

Chieh-Chun
Eurecom
RIC expert

Khoa
Eurecom
Data Scientist

Ian
NTUST-BMWLab
PhD student

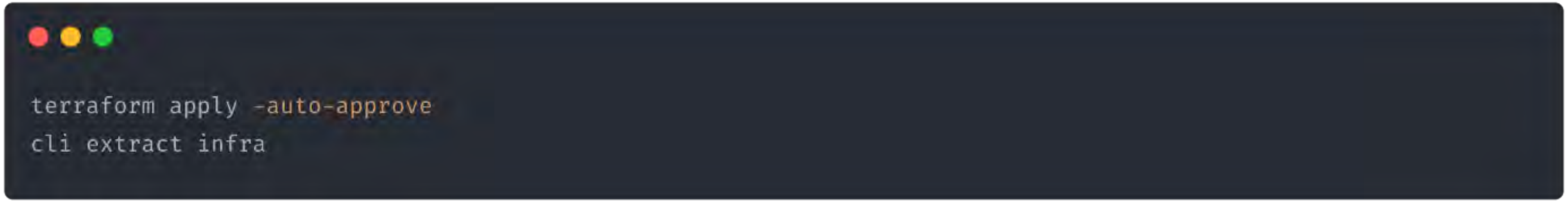
© BubbleRAN

§

Build, Operate, and Automate

A Cloud-native 5G Open RAN Network, ***Bottom Up***

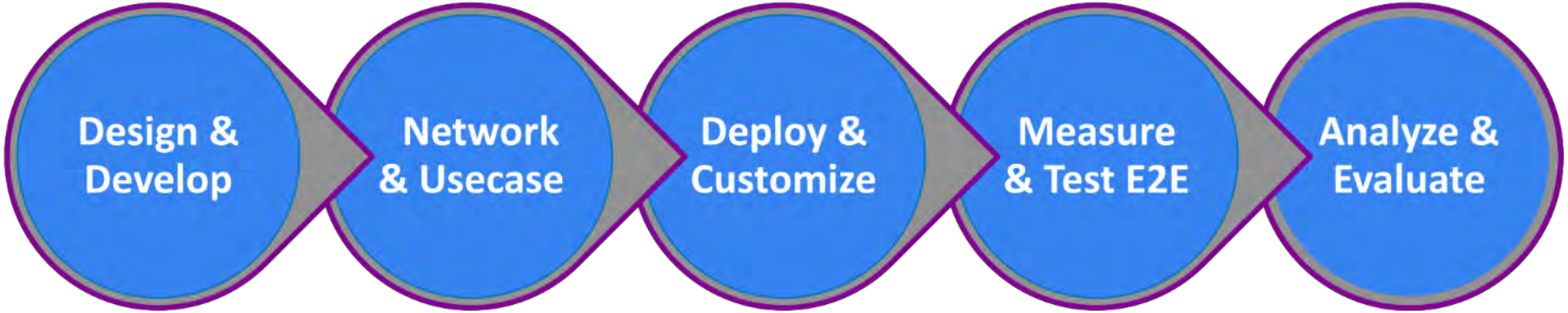
00 - Infrastructure



```
terraform apply -auto-approve
cli extract infra
```

- ★ Automated cluster deployment in one command
- ★ From infrastructure to ORS service, in just 10 minutes, with Pay-as-you-Go
- ★ Deployment of ORS over Google Kubernetes Engine (GKE) via Terraform

** - ORS in Nutshell



Open RAN Studio is world-first production-grade cloud-native platform to seamlessly design, operate, experiment an emulated end-to-end 3GPP & O-RAN standard-compliant network with edge services, at scale.

** - ORS features

- ★ Multi-infrastructure support: Bare-metal, On-premise, Public (GKE), Single-node (Microk8s)
- ★ O-RAN compliant SMO and RIC stack including Non-RT RIC, Near-RT RIC, and OAM
- ★ Network design, protocol tracing, log extraction, integrated UE testing
- ★ rApps and xApps for monitoring and control
- ★ End-to-end agile and scalable declarative deployment including UE
- ★ Day-2 features, including network reconfiguration, upgrade, and fault management
- ★ Difference between declarative and imperative deployments
- ★ Multi-vendor support: OAI, SRS, and Open5GS (both LTE and NR)
- ★ Programmable cloud-native observability with Grafana dashboard
- ★ Multi-source data lake, including RAN, Energy, and Infrastructure metrics

© BubbleRAN

** - ORS features

Bronze Release (v2.0) is here!

Rollout for current customers starting this week (Week #44)

New customers from December

Installation options available:

- Pay-as-you-Go ⇒ Google Cloud via Terraform
- Small Scale, Single Node ⇒ Microk8s
- Remote installation, Large Scale ⇒ Kubeadm and Cloud-init

** - ORS use cases

★ Education and training

★ Data collection and model training

★ rApp or xApp design and analysis

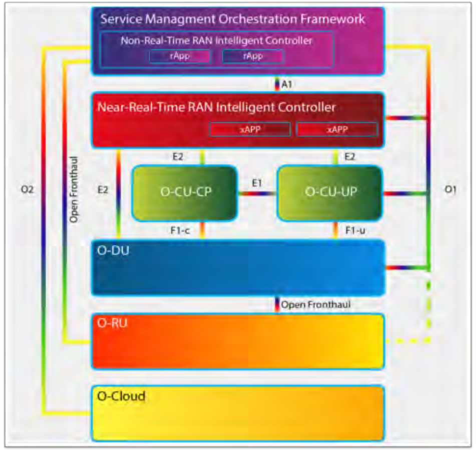
★ Interoperability testing

★ Test and measurements with UE in the loop

★ Research validation

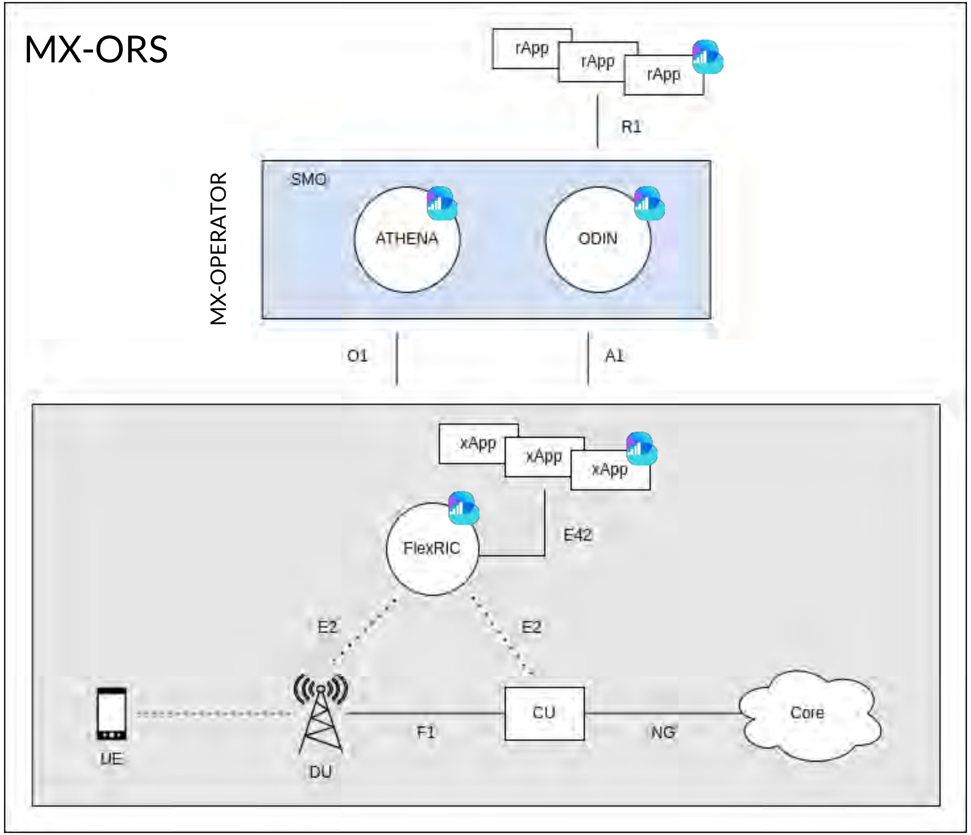
★ Network simulation and emulation

** - Revisiting the O-RAN Ecosystem

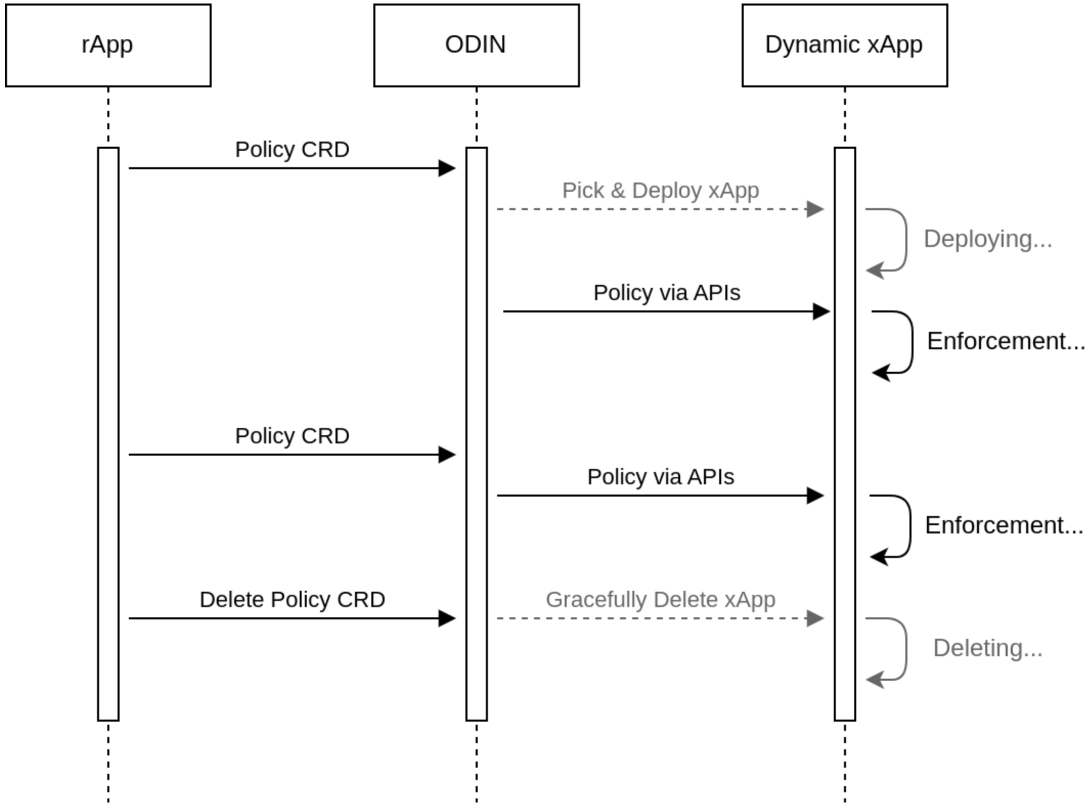

Service Managment Orchestration Framework
Non-Real-Time RAN Intelligent Controller
rApp
rApp
A1
Near-Real-Time RAN Intelligent Controller
xAPP
xAPP
E2
E2
O2
Open Fronthaul
E2
O-CU-CP
E1
O-CU-UP
O1
F1-c
F1-u
O-DU
Open Fronthaul
O-RU
O-Cloud

O1: SMO to all
O2: SMO to O-Cloud
A1: Non-RT RIC to Neal-RT RIC
E1: O-CU UP and CP
E2: Neal-RT RIC to E2 Nodes
F1-C: O-CU CP to O-DU CP
F1-U: O-CU UP to O-DU UP
Open Fronthaul(7.2): O-DU to O-RU

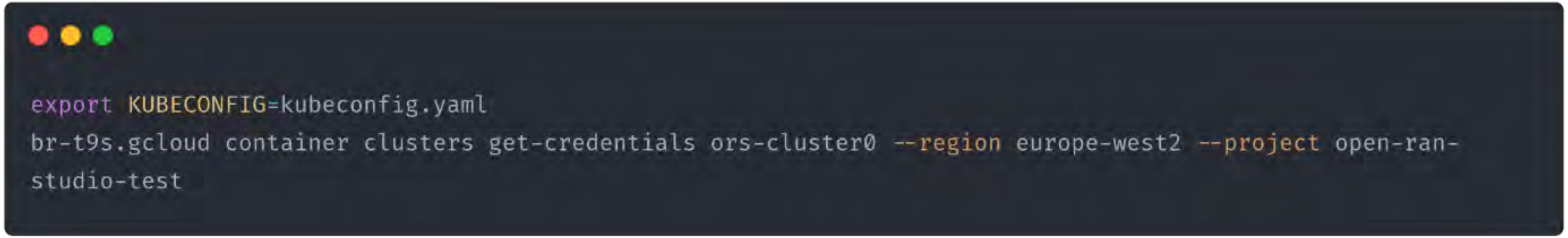
ODIN = Observable Distributed Intelligent Networking



** - Revisiting the O-RAN Ecosystem



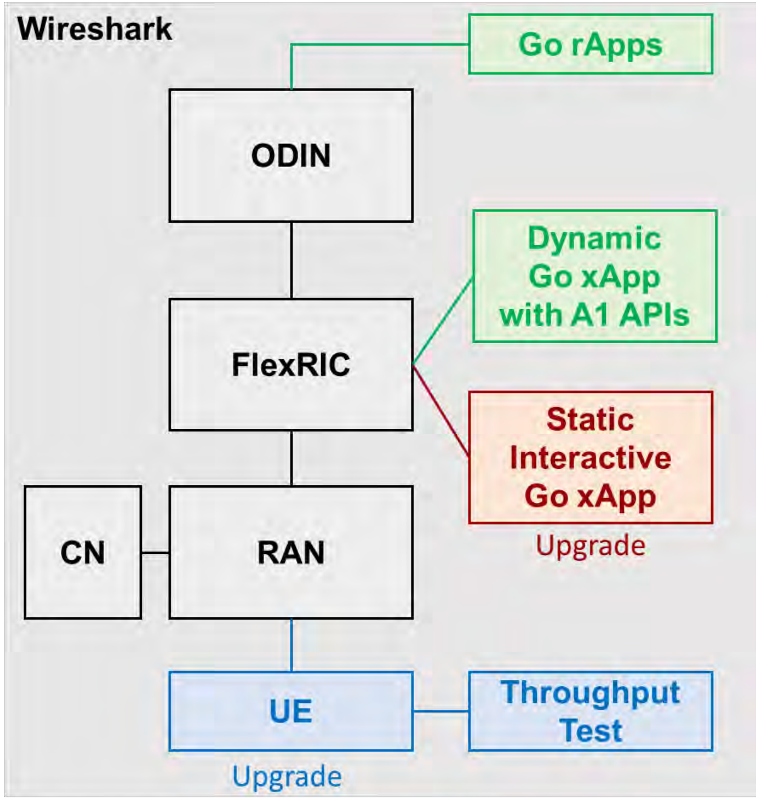
** - GKE Authentication



```
export KUBECONFIG=kubeconfig.yaml
br-t9s.gcloud container clusters get-credentials ors-cluster0 --region europe-west2 --project open-ran-
studio-test
```

- ★ Get the config generated from Terraform
- ★ Or manually download the config from gcloud

** - Deploy network step by step

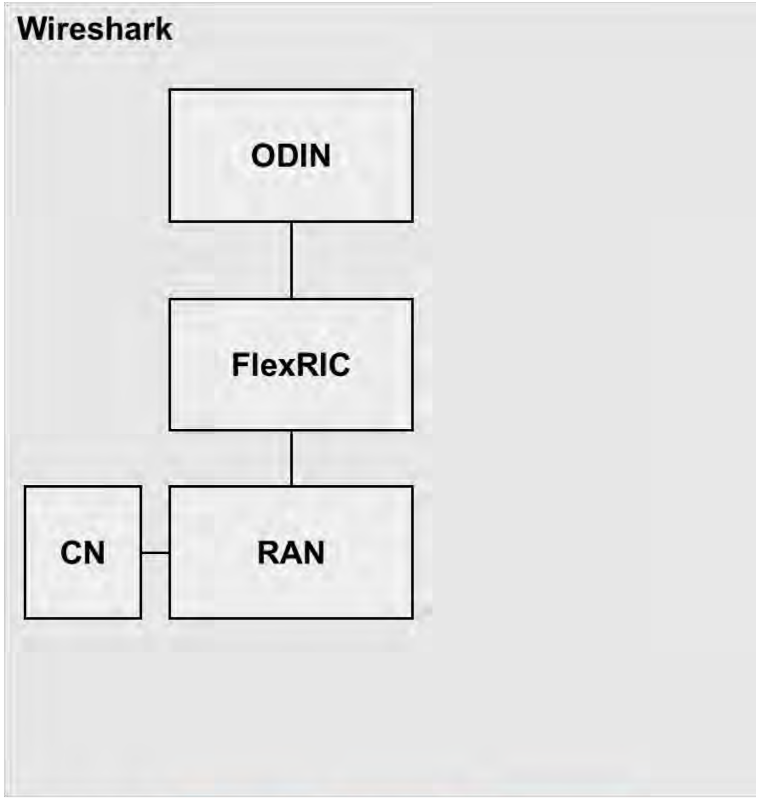


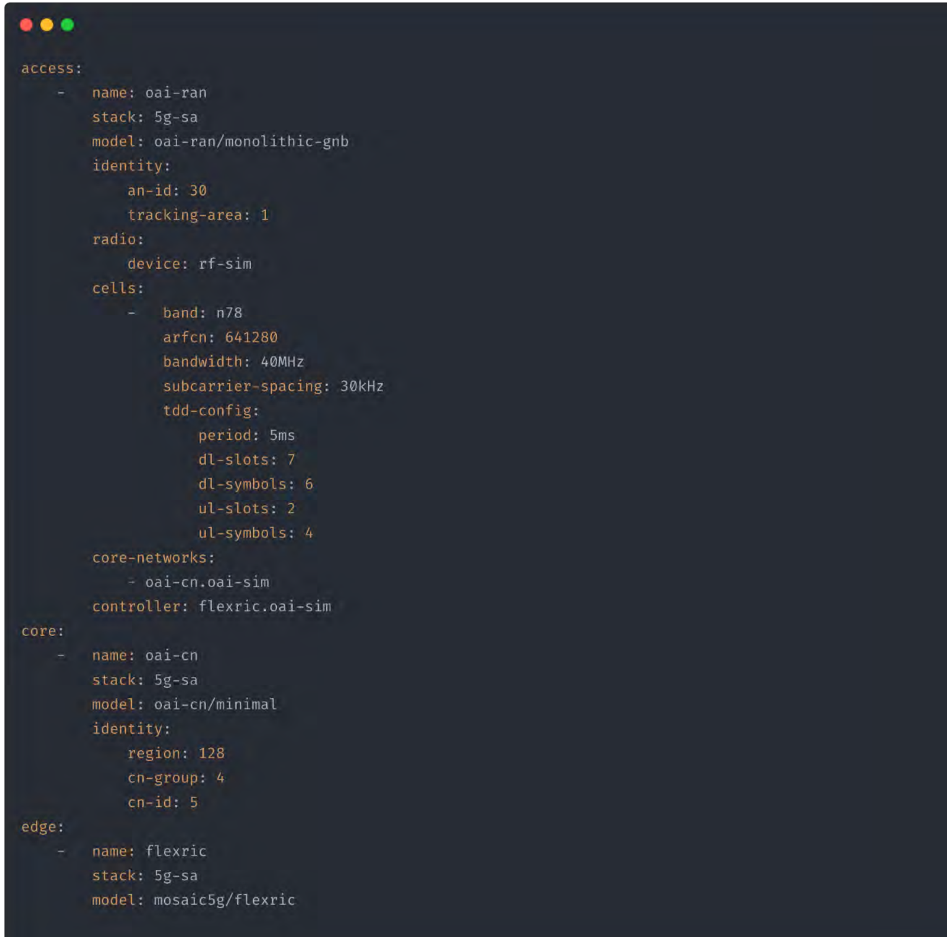
Step1 | Step 2 | Step 3 | Step 4

- ★ Design a simple network with O-RAN stack
- ★ Step-by-step design
- ★ Day-2 operations
- ★ Test and measurement
- ★ Packet tracing
- ★ xApps and rApps
- ★ UE in the loop

© BubbleRAN

01 - Step-1 Network



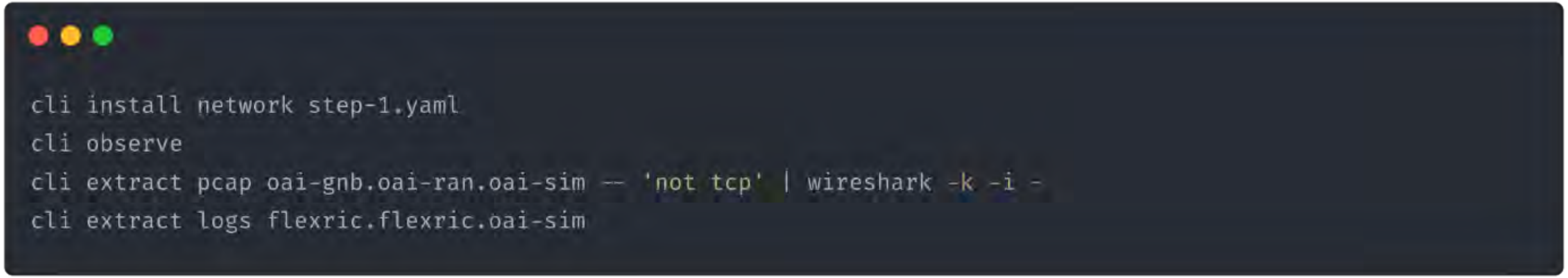


```
access:
    -   name: oai-ran
        stack: 5g-sa
        model: oai-ran/monolithic-gnb
        identity:
            an-id: 30
            tracking-area: 1
        radio:
            device: rf-sim
        cells:
            -   band: n78
                arfcn: 641280
                bandwidth: 40MHz
                subcarrier-spacing: 30kHz
                tdd-config:
                    period: 5ms
                    dl-slots: 7
                    dl-symbols: 6
                    ul-slots: 2
                    ul-symbols: 4
        core-networks:
            - oai-cn.oai-sim
        controller: flexric.oai-sim
core:
    -   name: oai-cn
        stack: 5g-sa
        model: oai-cn/minimal
        identity:
            region: 128
            cn-group: 4
            cn-id: 5
edge:
    -   name: flexric
        stack: 5g-sa
        model: mosaic5g/flexric
```

Step1 Step 2 Step 3 Step 4

© BubbleRAN

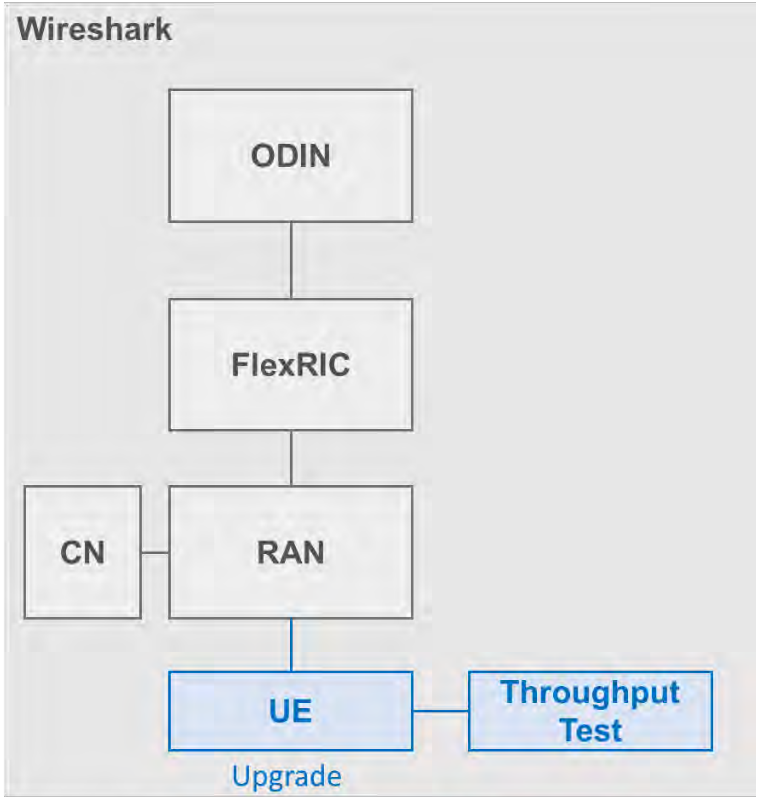
01 - Step-1 Network



```
cli install network step-1.yaml
cli observe
cli extract pcap oai-gnb.oai-ran.oai-sim -- 'not tcp' | wireshark -k -i -
cli extract logs flexric.flexric.oai-sim
```

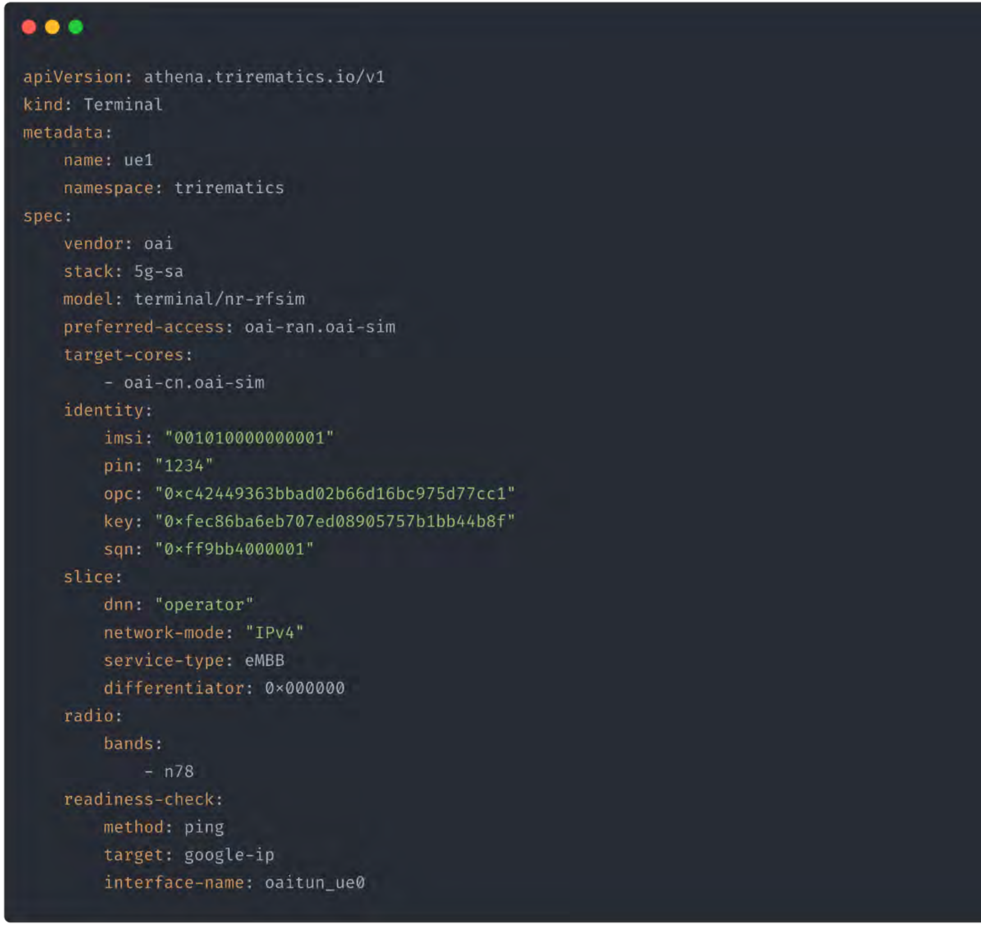
- ★ OAI gNB 5G-SA
- ★ OAI 5GC
- ★ BR-FlexRIC [FlexRIC with enhanced functionality]
- ★ Wireshark trace for following the UE messages later
- ★ Log extraction to verify E2 connection

02 - Step-2 Network



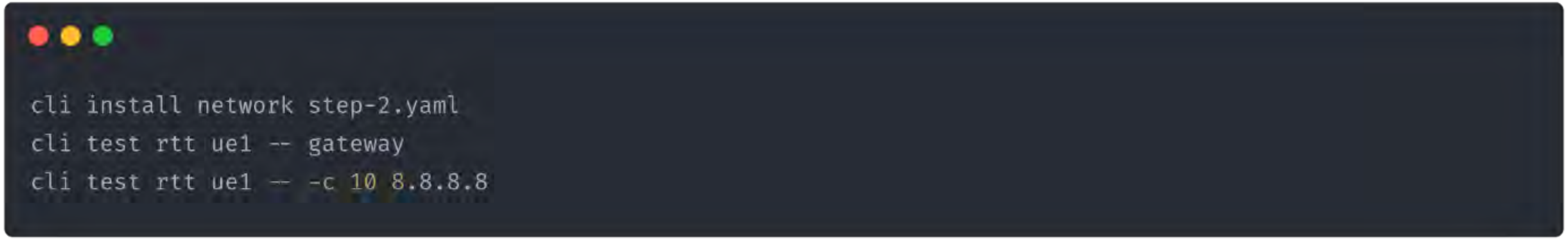
Step1 Step 2 Step 3 Step 4

© BubbleRAN



```yaml
apiVersion: athena.trirematics.io/v1
kind: Terminal
metadata:
    name: ue1
    namespace: trirematics
spec:
    vendor: oai
    stack: 5g-sa
    model: terminal/nr-rfsim
    preferred-access: oai-ran.oai-sim
    target-cores:
        - oai-cn.oai-sim
    identity:
        imsi: "001010000000001"
        pin: "1234"
        opc: "0xc42449363bbad02b66d16bc975d77cc1"
        key: "0xfec86ba6eb707ed08905757b1bb44b8f"
        sqn: "0xff9bb4000001"
    slice:
        dnn: "operator"
        network-mode: "IPv4"
        service-type: eMBB
        differentiator: 0x000000
    radio:
        bands:
            - n78
    readiness-check:
        method: ping
        target: google-ip
        interface-name: oaitun_ue0
```

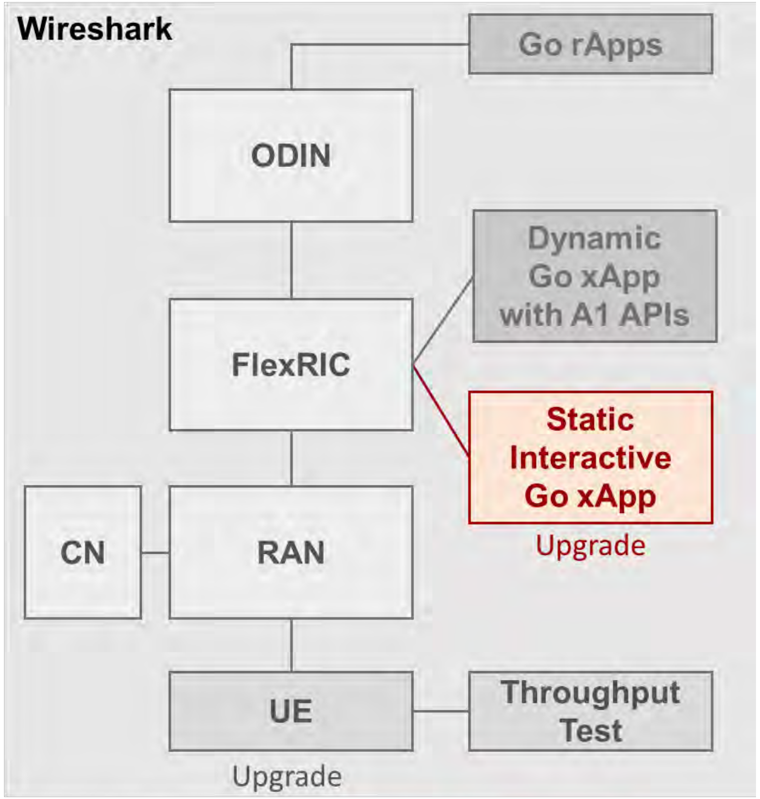
02 - Step-2 Network

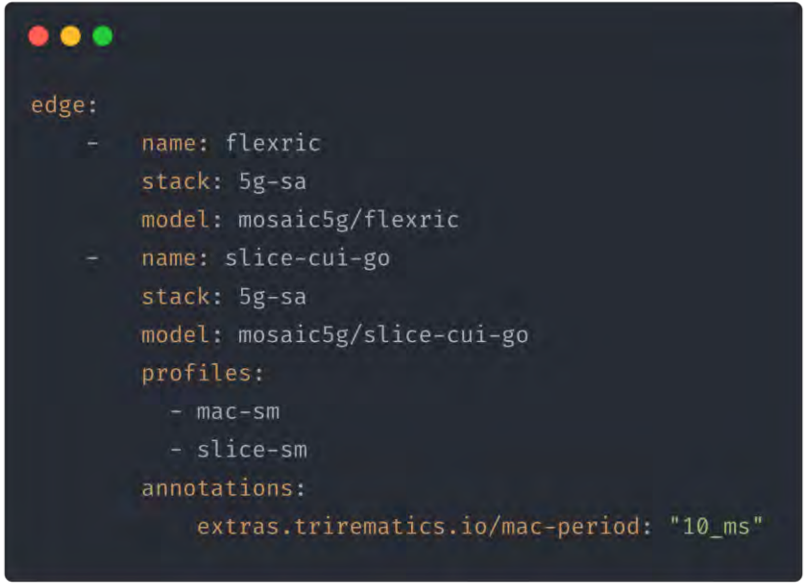


```
cli install network step-2.yaml
cli test rtt ue1 -- gateway
cli test rtt ue1 -- -c 10 8.8.8.8
```

- ★ Add UE in the loop
- ★ Checkout the UE messages in Wireshark
- ★ RTT test with the UE

03 - Step-3 Network

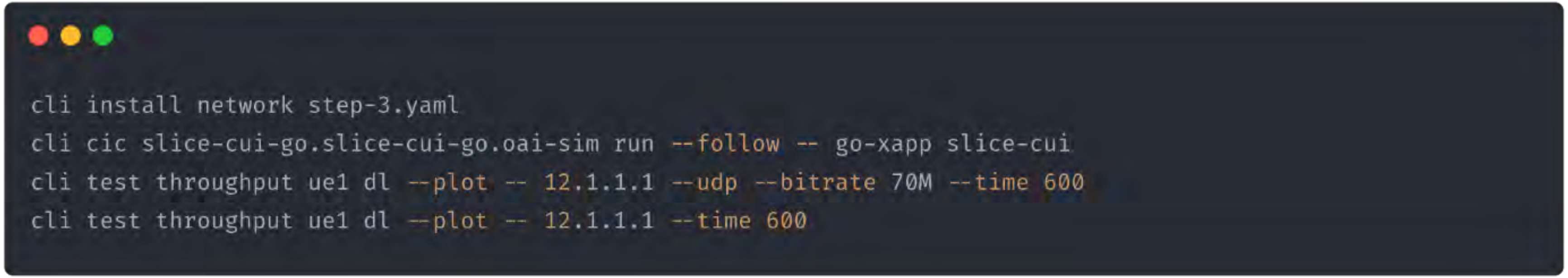




```
edge:
    - name: flexric
      stack: 5g-sa
      model: mosaic5g/flexric
    - name: slice-cui-go
      stack: 5g-sa
      model: mosaic5g/slice-cui-go
      profiles:
        - mac-sm
        - slice-sm
      annotations:
        extras.trirematics.io/mac-period: "10_ms"
```

Step1 Step 2 Step 3 Step 4

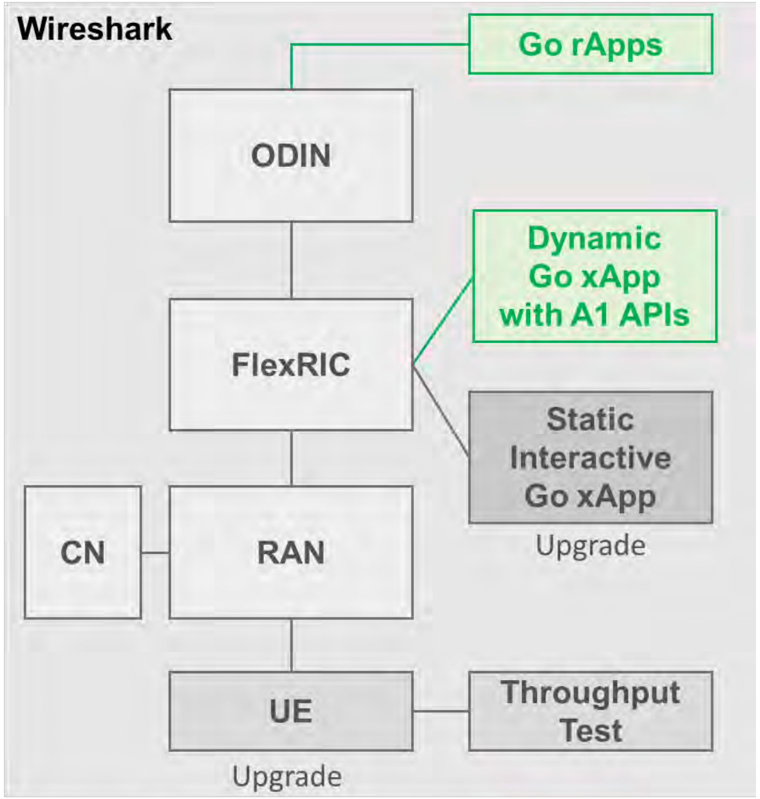
03 - Step-3 Network

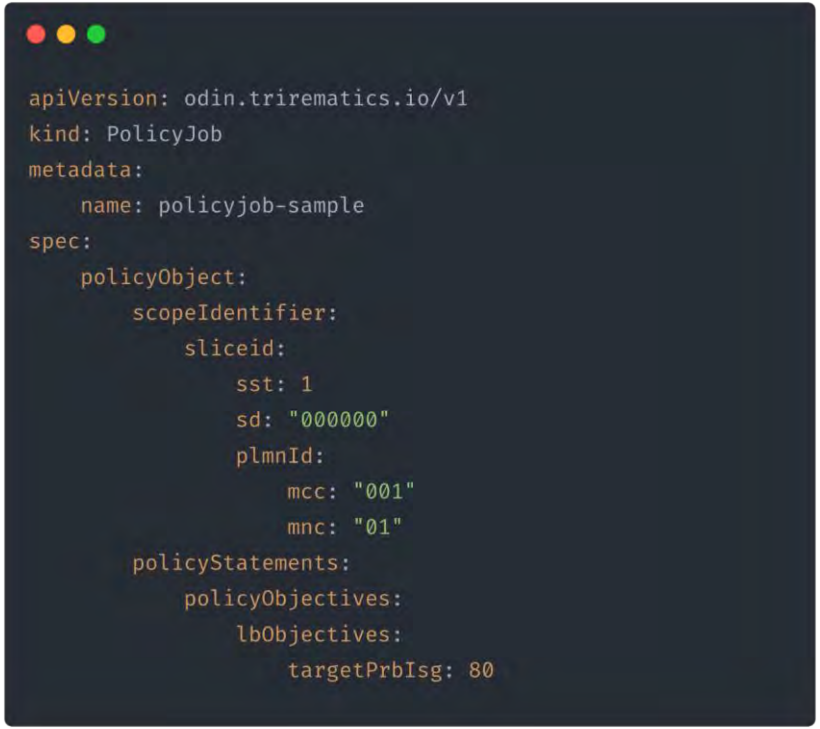


```
cli install network step-3.yaml
cli cic slice-cui-go.slice-cui-go.oai-sim run --follow -- go-xapp slice-cui
cli test throughput ue1 dl --plot -- 12.1.1.1 --udp --bitrate 70M --time 600
cli test throughput ue1 dl --plot -- 12.1.1.1 --time 600
```

- ★ Day-2: Add xApp
- ★ Open access to interactive xApp
- ★ Run throughput tests with plotting
- ★ Check the PRB utilization from the xApp

04 - Step-4 Network

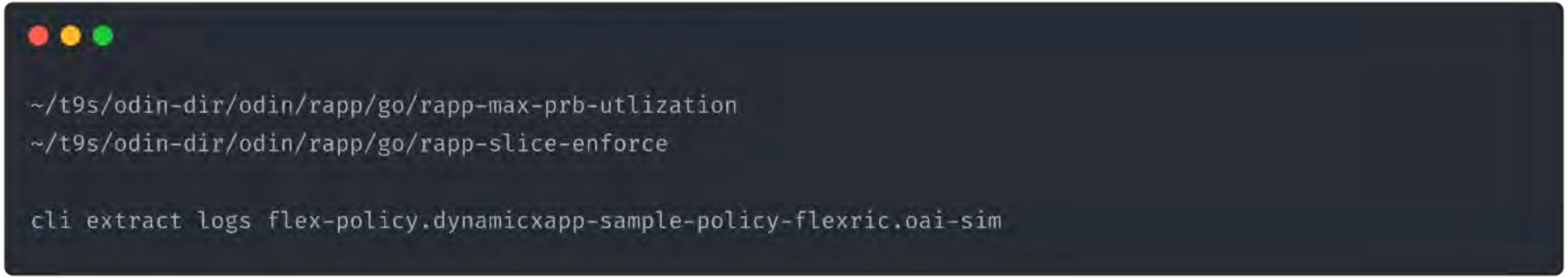




```yaml
apiVersion: odin.trirematics.io/v1
kind: PolicyJob
metadata:
    name: policyjob-sample
spec:
    policyObject:
        scopeIdentifier:
            sliceid:
                sst: 1
                sd: "000000"
                plmnId:
                    mcc: "001"
                    mnc: "01"
        policyStatements:
            policyObjectives:
                lbObjectives:
                    targetPrbIsg: 80
```

Step1 Step 2 Step 3 Step 4

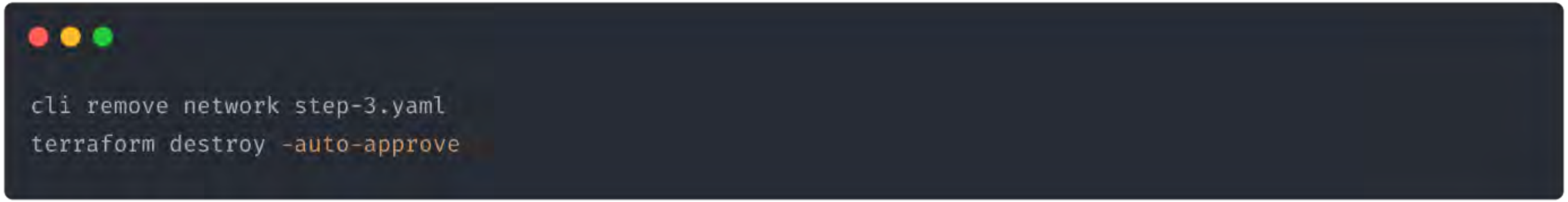
04 - Step-4 Network



```
~/t9s/odin-dir/odin/rapp/go/rapp-max-prb-utlization
~/t9s/odin-dir/odin/rapp/go/rapp-slice-enforce

cli extract logs flex-policy.dynamicxapp-sample-policy-flexric.oai-sim
```

- ★ rApps and dynamic xApps
- ★ Policy 1 → Maximum PRB Utilization ⇒ Control Action 1 → RAN Slicing
- ★ Policy 2 → Slice Enforcement ⇒ Control Action 2 → RAN Slicing, User Association

** - Step-∞ Destroy



```
cli remove network step-3.yaml
terraform destroy -auto-approve
```

- ★ Remove the network
- ★ Release the infrastructure
- ★ View the costs

Short break

5 minutes

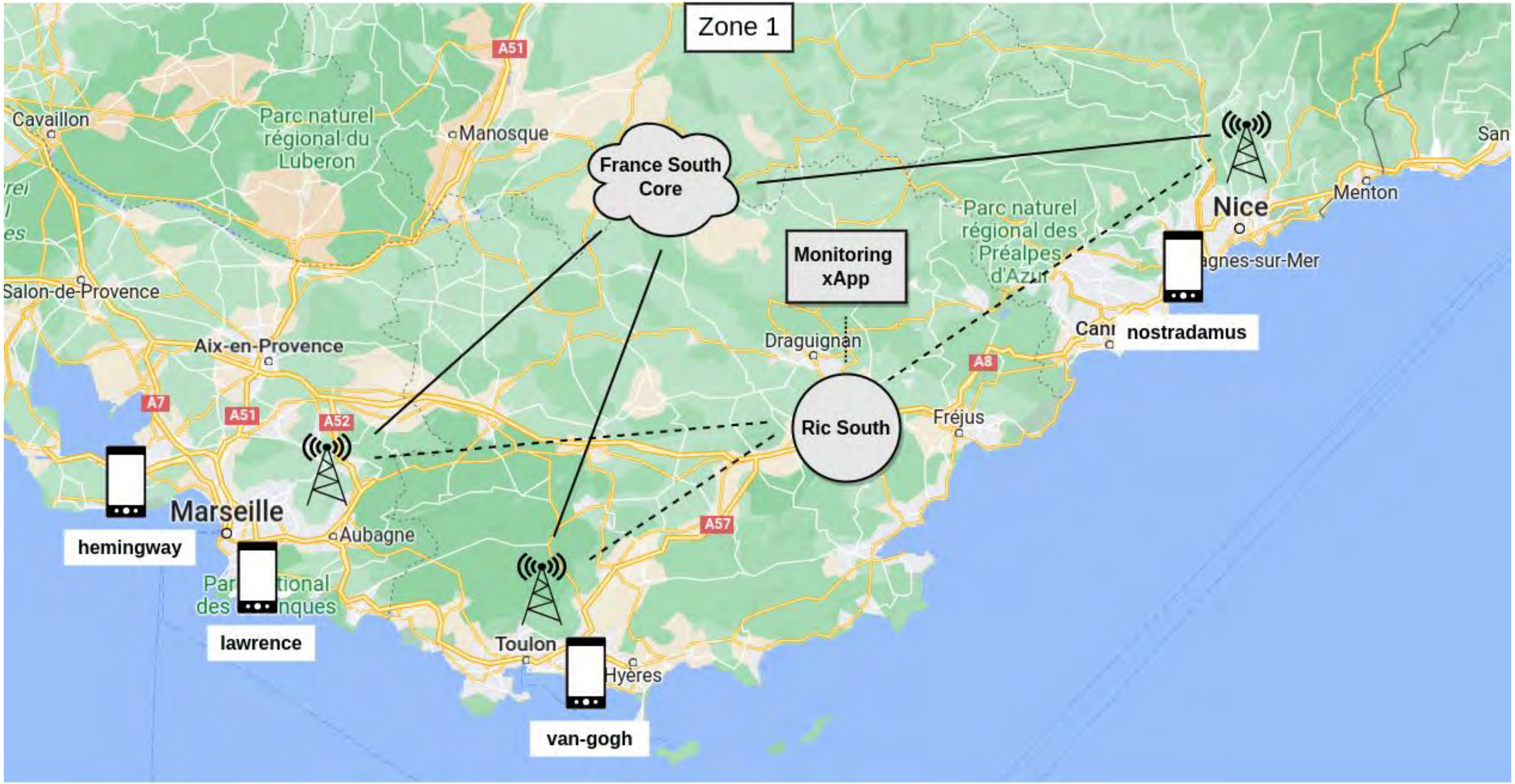
© BubbleRAN

§

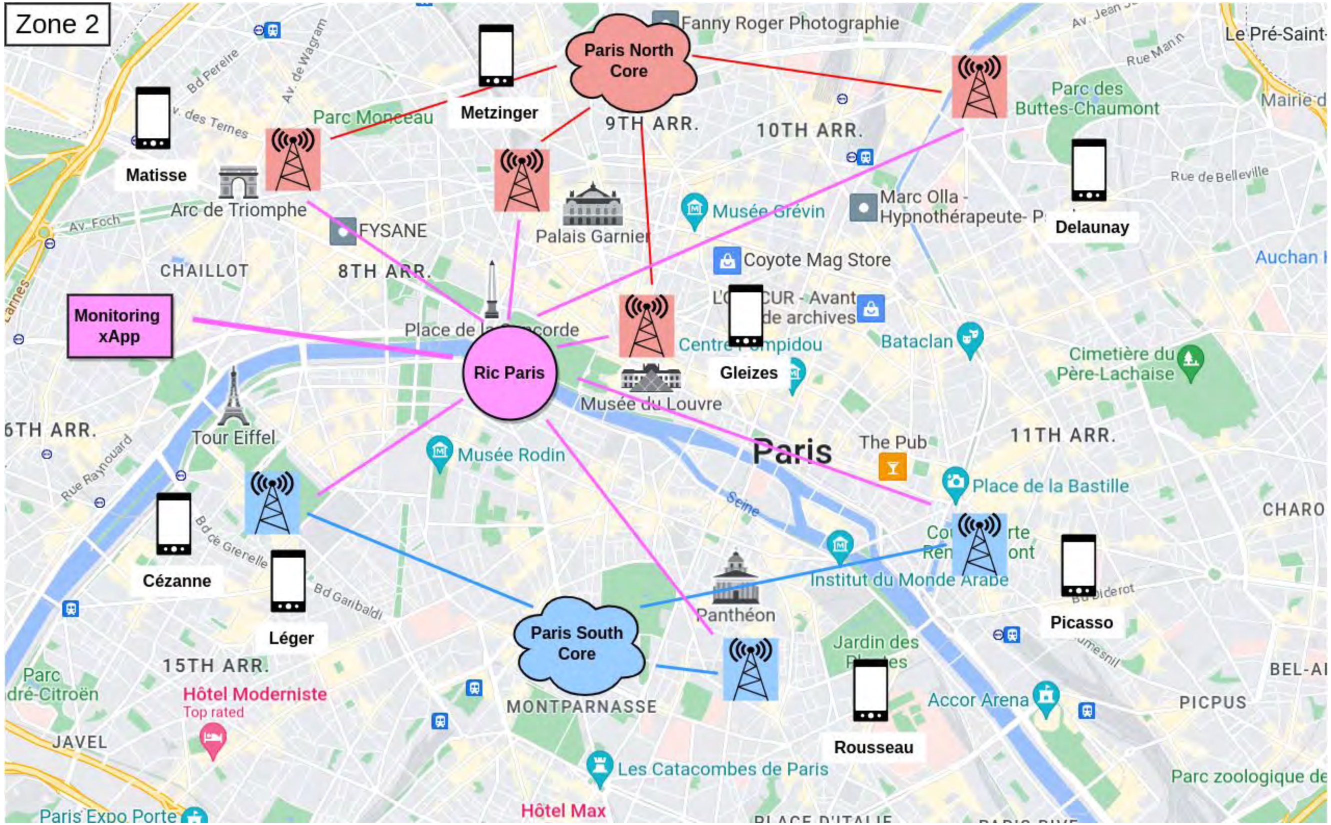
Large Scale Open RAN

ORS Observability for large scale networks

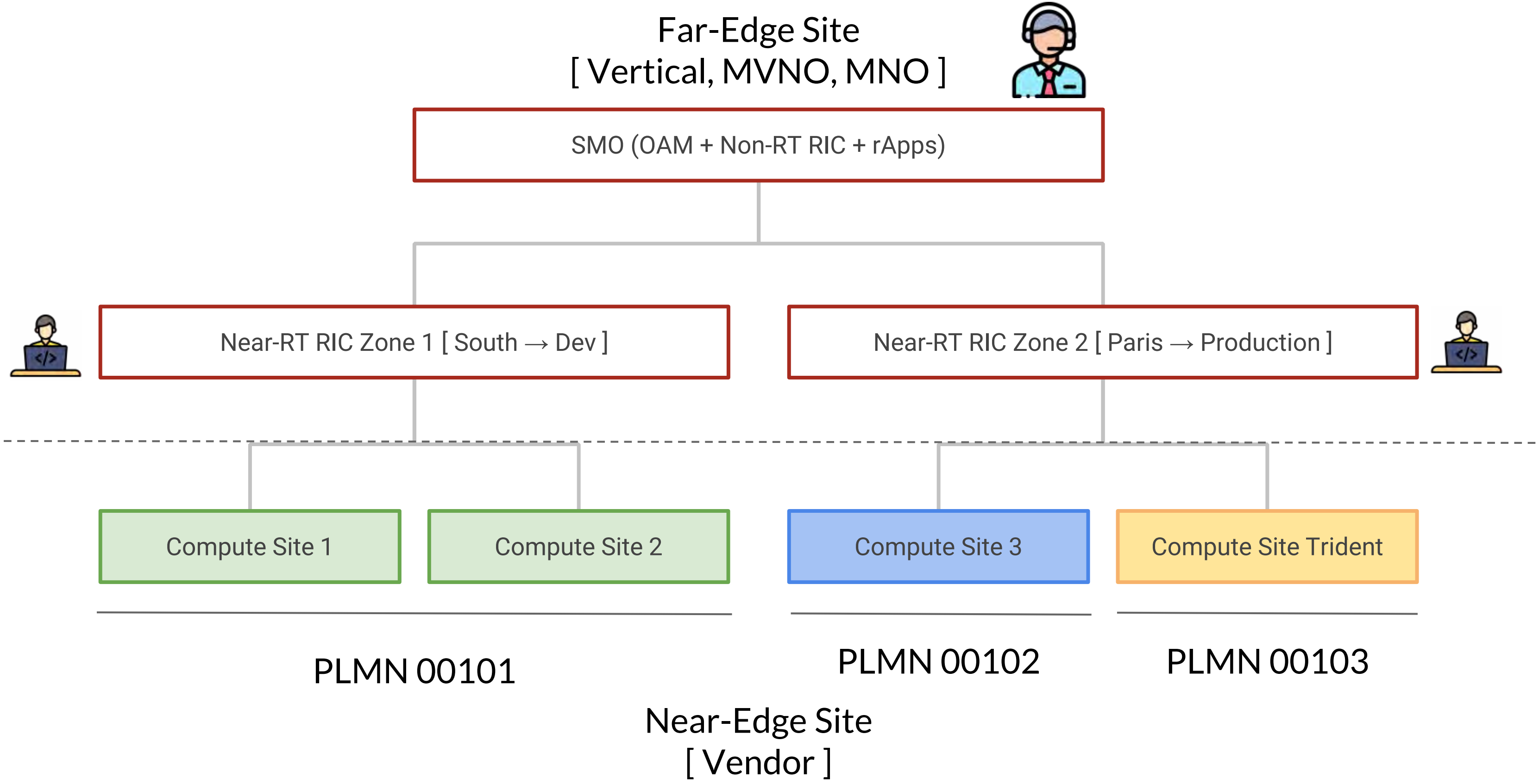
Storyline


Zone 1
France South Core
Monitoring xApp
Ric South
nostradamus
hemingway
lawrence
van-gogh
Cavaillon
Parc naturel régional du Luberon
Manosque
Salon-de-Provence
Aix-en-Provence
Marseille
Aubagne
Toulon
Hyères
Draguignan
Fréjus
Parc naturel régional des Préalpes d'Azur
Nice
Menton
A51
A7
A52
A57
A8

Storyline


Zone 2
Paris North Core
Metzinger
Matisse
Delaunay
Monitoring xApp
Ric Paris
Gleizes
Cézanne
Léger
Picasso
Rousseau
Paris South Core
Paris
Arc de Triomphe
Palais Garnier
Musée du Louvre
Tour Eiffel
Musée Rodin
Panthéon
Place de la Bastille
Cimetière du Père-Lachaise
Parc des Buttes-Chaumont
Les Catacombes de Paris
9TH ARR.
10TH ARR.
8TH ARR.
11TH ARR.
15TH ARR.
MONTPARNASSE
CHAILLOT
Seine

Storyline


Far-Edge Site
[Vertical, MVNO, MNO]
SMO (OAM + Non-RT RIC + rApps)
Near-RT RIC Zone 1 [South → Dev]
Near-RT RIC Zone 2 [Paris → Production]
Compute Site 1
Compute Site 2
Compute Site 3
Compute Site Trident
PLMN 00101
PLMN 00102
PLMN 00103
Near-Edge Site
[Vendor]

Statistics

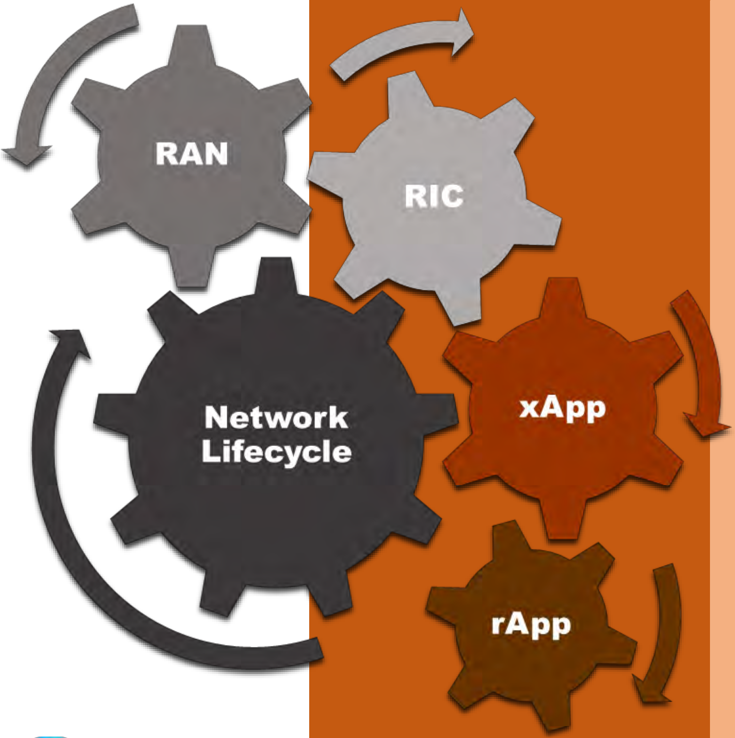
Containers	+200
Machines	10
RANs	10
UEs	12
Total CPUs [not allocated]	194
Time	<2 min

Open RAN enables management and ***control*** *of such large networks by utilizing different range of* ***xApps*** *and* ***rApps***

© BubbleRAN

Open RAN Ecosystem

xApp ? rApp



Independent xApp Lifecycle

- Accelerating innovative xApp development
- Operating xApp dynamically through SMO
- Leading the way of xApp evolution in O-RAN ecosystem
- Bridging xApp and rApp seamlessly
- Enabling a smooth transition from RAN to data

© BubbleRAN

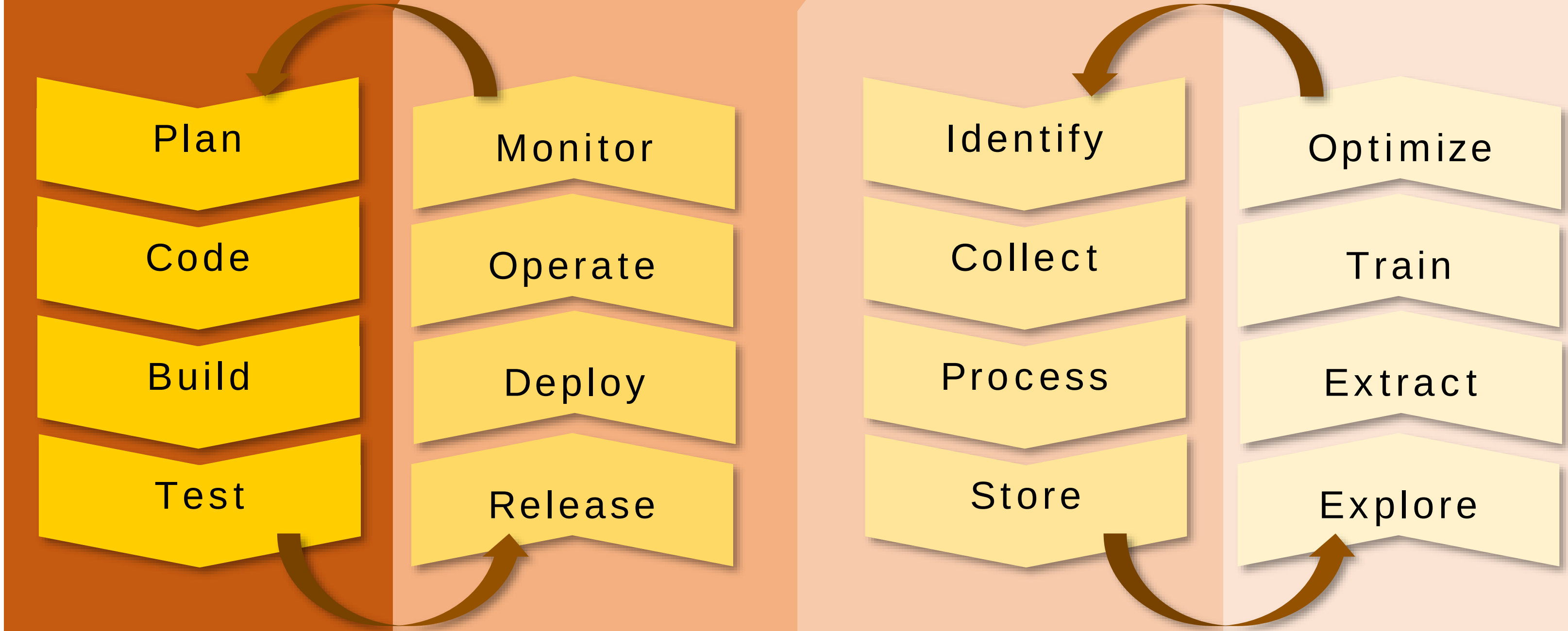
Open RAN Ecosystem

xApp DevOps

Binary xApp · **Containerized xApp**

Data & ML

Integrated xApp · **rApp**



	xApp DevOps		Data & ML	
Open RAN Ecosystem	Binary xApp	Containerized xApp	Integrated xApp	rApp
Roles	**Developer**	**Maintainer**	**Vertical User**	**Business intelligent & Data analyst**
Participants	**Vendor**	**Operator**	**Application Provider**	**Stakeholder**
Usage Scenario	**Research & Develop**	**Testing & Measurement**	**Production**	**Analysis & Optimize**
Knowledge of NearRT-RIC APIs	**Proficient**	**Moderate**	**Basic**	**None**
Knowledge of SMO APIs	**None**	**Basic**	**Moderate**	**Proficient**
Actions	**Develop new functionalities**	**Interact with network**	**Apply & Enforce policy**	**Define network intent & Create policy**
Programming Languages	C/C++, Python, Go	Python, Go	Python, Go	Any
Network Configuration	Manual	Automatic	Automatic	Automatic
Network Deployment	Static	Static	Dynamic	Dynamic
Demo Example	C xApp with monitor service	Python & Go xApp with interactive module	Go xApp with A1 APIs & Programmable Python xApp	Coming Soon

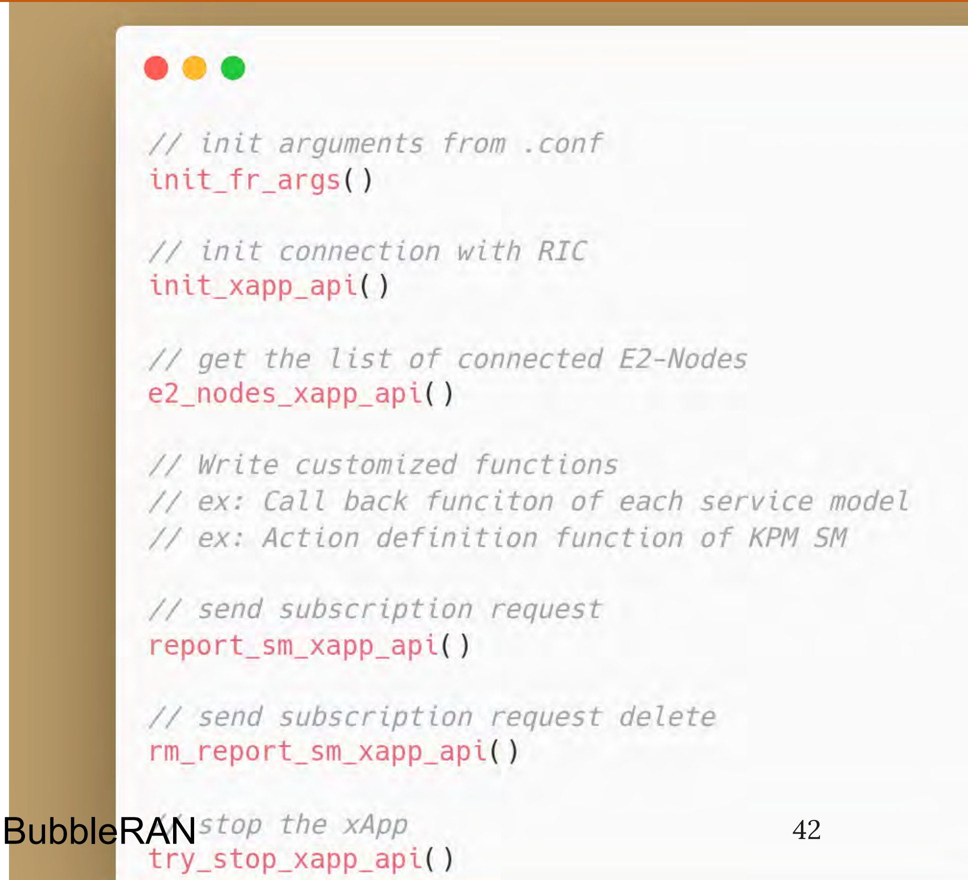
	xApp DevOps		Data & ML	
Open RAN Ecosystem	**Binary xApp**	**Containerized xApp**	**Integrated xApp**	**rApp**
Roles	**Developer**	**Maintainer**	**Vertical User**	**Business intelligent & Data analyst**
Participants	**Vendor**	**Operator**	**Application Provider**	**Stakeholder**
Usage Scenario	**Research & Develop**	**Testing & Measurement**	**Production**	**Analysis & Optimize**
Knowledge of NearRT-RIC APIs	**Proficient**	**Moderate**	**Basic**	**None**
Knowledge of SMO APIs	**None**	**Basic**	**Moderate**	**Proficient**
Actions	**Develop new functionalities**	**Interact with network**	**Apply & Enforce policy**	**Define network intent & Create policy**
Programming Languages	C/C++, Python, Go	Python, Go	Python, Go	Any
Network Configuration	Manual	Automatic	Automatic	Automatic
Network Deployment	Static	Static	Dynamic	Dynamic
	C xApp with monitor service	Python & Go xApp with interactive module	Go xApp with A1 APIs & Programmable Python xApp	Coming Soon

© BubbleRAN

CopyRight BubbleRAN

Binary xApp
C xApp with monitor service

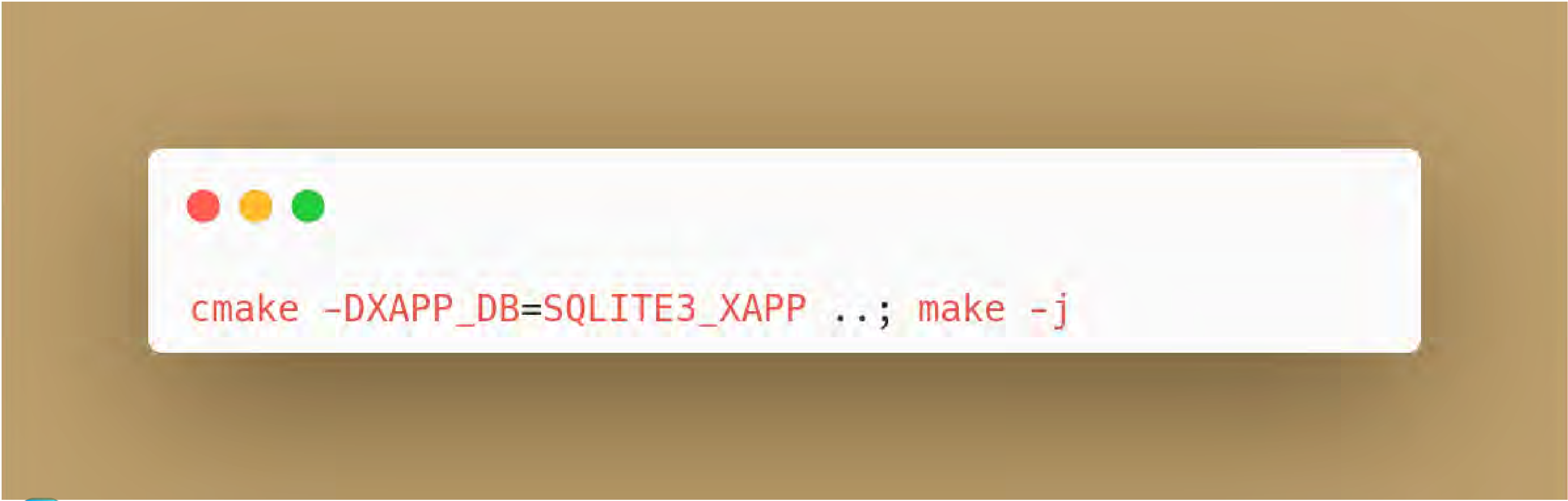
1. Develop xapp.c by using NearRT-RIC APIs provided by FlexRIC



```
// init arguments from .conf
init_fr_args()

// init connection with RIC
init_xapp_api()

// get the list of connected E2-Nodes
e2_nodes_xapp_api()

// Write customized functions
// ex: Call back funciton of each service model
// ex: Action definition function of KPM SM

// send subscription request
report_sm_xapp_api()

// send subscription request delete
rm_report_sm_xapp_api()

// stop the xApp
try_stop_xapp_api()
```

Binary xApp
C xApp with monitor service

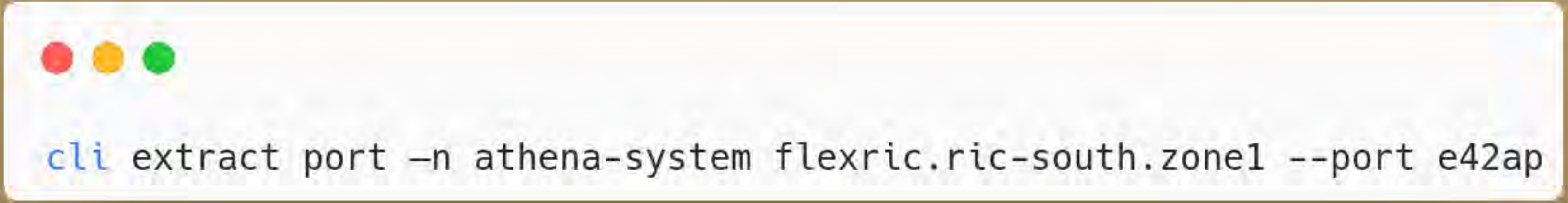
2. Build xapp.c with SQLite3



```
cmake -DXAPP_DB=SQLITE3_XAPP ..; make -j
```

Binary xApp
C xApp with monitor service

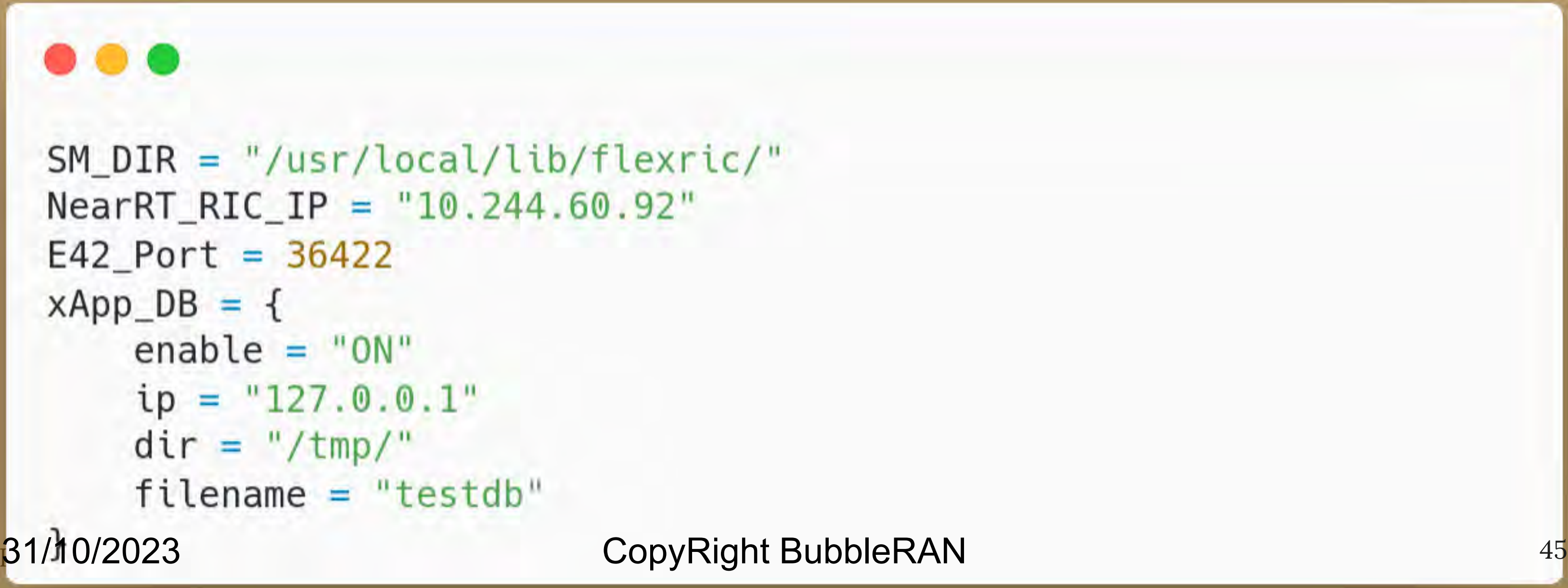
3. Get IP address and port number of existing RIC from the cluster



```
cli extract port –n athena-system flexric.ric-south.zone1 --port e42ap
```

Binary xApp
C xApp with monitor service

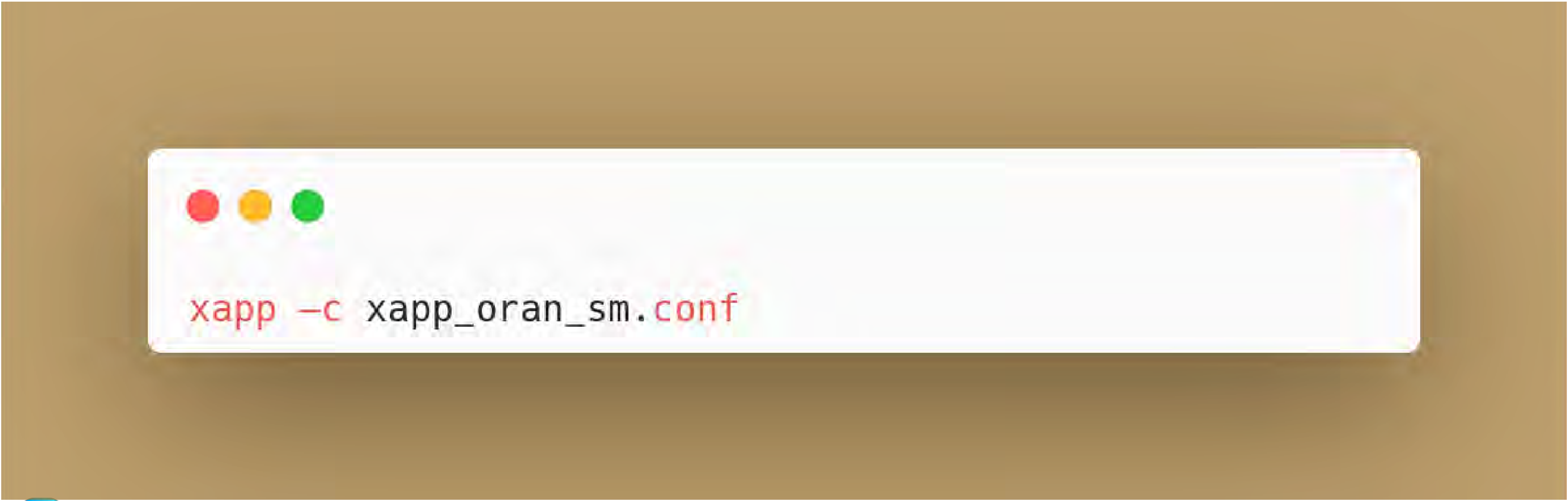
4. Configure xapp_oran_sm.conf



```
SM_DIR = "/usr/local/lib/flexric/"
NearRT_RIC_IP = "10.244.60.92"
E42_Port = 36422
xApp_DB = {
    enable = "ON"
    ip = "127.0.0.1"
    dir = "/tmp/"
    filename = "testdb"
}
```

Binary xApp
C xApp with monitor service

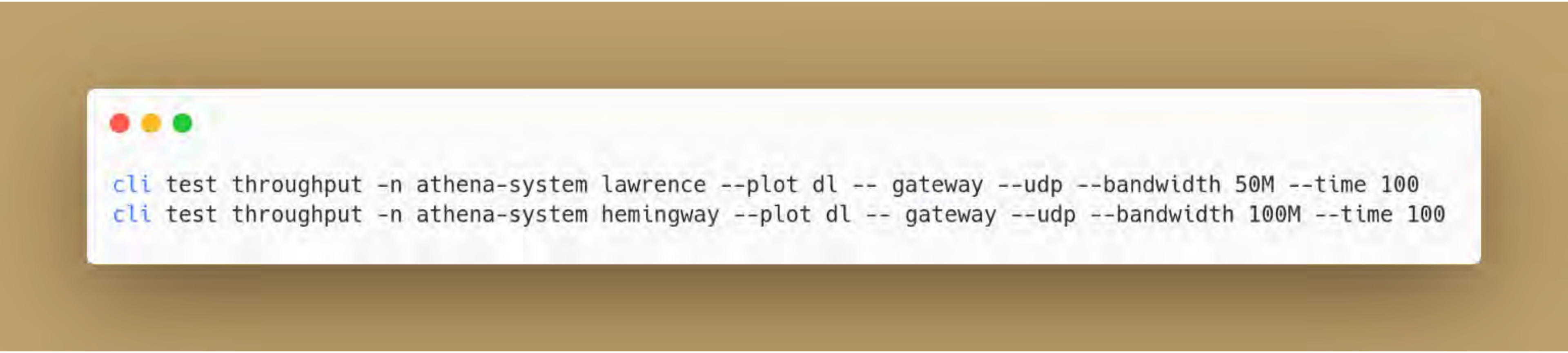
5. Run xApp



```
xapp -c xapp_oran_sm.conf
```

Binary xApp
C xApp with monitor service

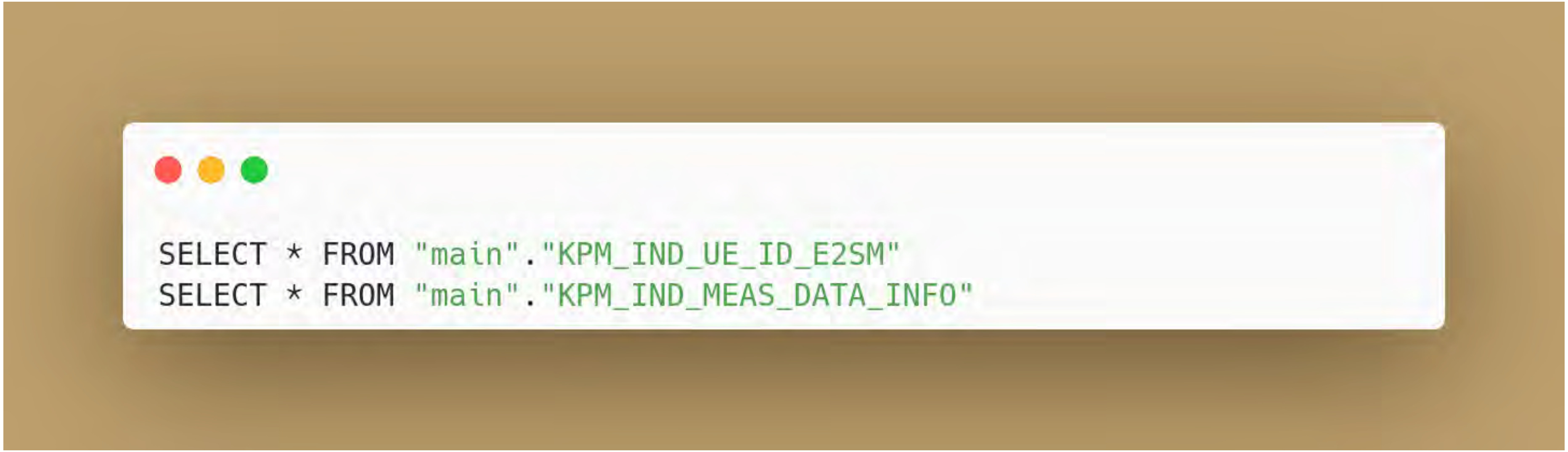
6. Run UEs throughput test



```
cli test throughput -n athena-system lawrence --plot dl -- gateway --udp --bandwidth 50M --time 100
cli test throughput -n athena-system hemingway --plot dl -- gateway --udp --bandwidth 100M --time 100
```

Binary xApp
C xApp with monitor service

7. Open SQLite3 database to query the collected KPM data



```
SELECT * FROM "main"."KPM_IND_UE_ID_E2SM"
SELECT * FROM "main"."KPM_IND_MEAS_DATA_INFO"
```

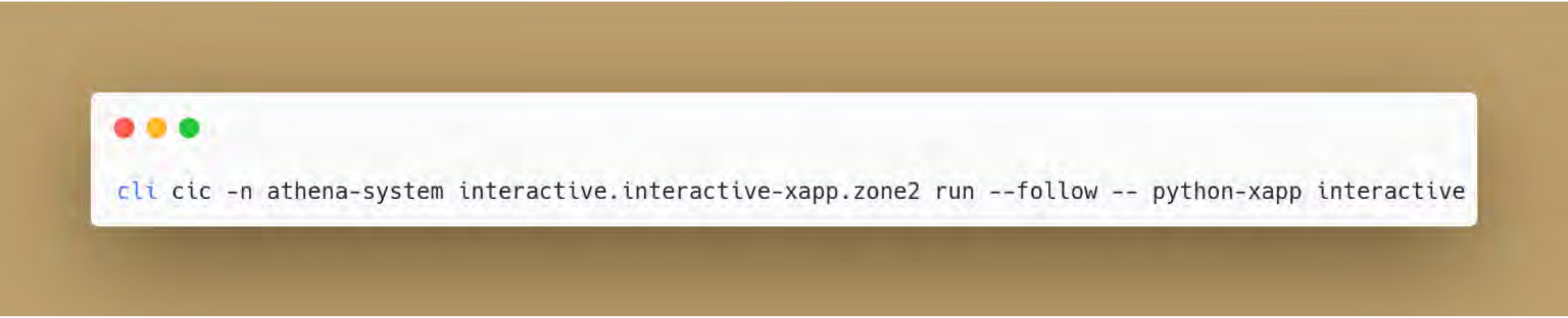
	xApp DevOps		Data & ML	
Open RAN Ecosystem	**Binary xApp**	**Containerized xApp**	**Integrated xApp**	**rApp**
Roles	**Developer**	**Maintainer**	**Vertical User**	**Business intelligent & Data analyst**
Participants	**Vendor**	**Operator**	**Application Provider**	**Stakeholder**
Usage Scenario	**Research & Develop**	**Testing & Measurement**	**Production**	**Analysis & Optimize**
Knowledge of NearRT-RIC APIs	**Proficient**	**Moderate**	**Basic**	**None**
Knowledge of SMO APIs	**None**	**Basic**	**Moderate**	**Proficient**
Actions	**Develop new functionalities**	**Interact with network**	**Apply & Enforce policy**	**Define network intent & Create policy**
Programming Languages	C/C++, Python, Go	Python, Go	Python, Go	Any
Network Configuration	Manual	Automatic	Automatic	Automatic
Network Deployment	Static	Static	Dynamic	Dynamic
Demo Example	C xApp with monitor service	Python & Go xApp with interactive module	Go xApp with A1 APIs & Programmable Python xApp	Coming Soon

© BubbleRAN

CopyRight BubbleRAN

Containerized xApp
Python xApp with interactive module

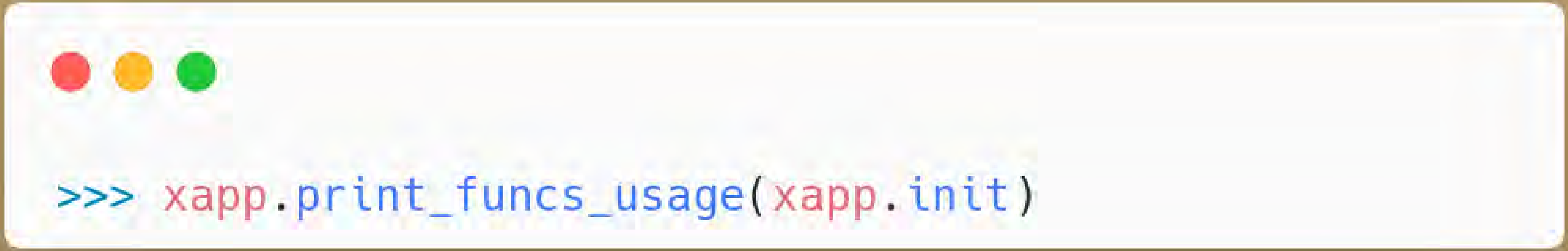
1. Access to deployed running xApp



```
cli cic -n athena-system interactive.interactive-xapp.zone2 run --follow -- python-xapp interactive
```

Containerized xApp
Python xApp with interactive module

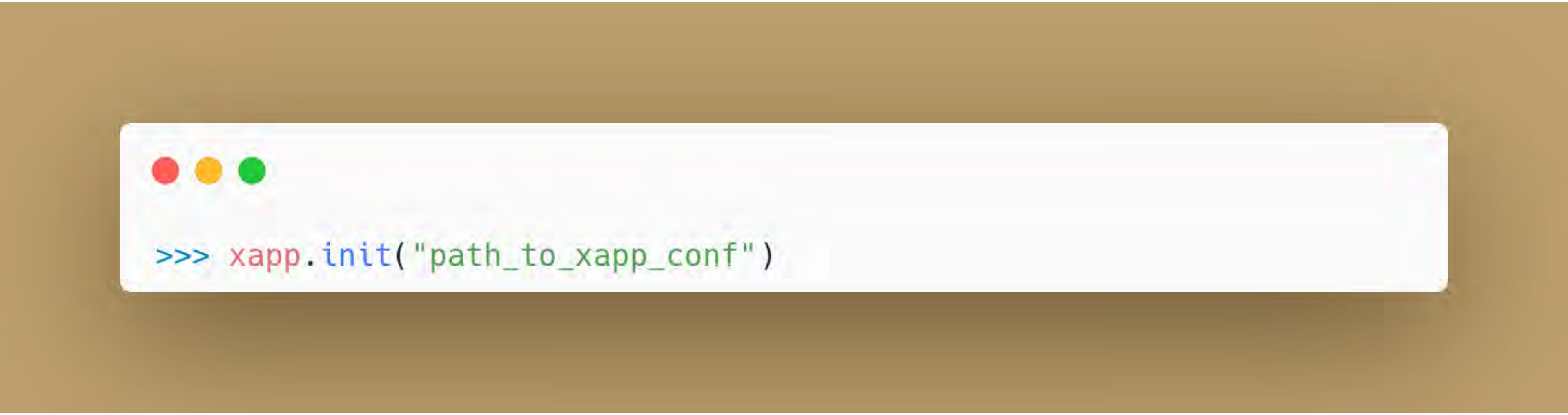
1. Show the functionalities provided by xApp to do the testing



```
>>> xapp.print_funcs_usage(xapp.init)
```

Containerized xApp
Python xApp with interactive module

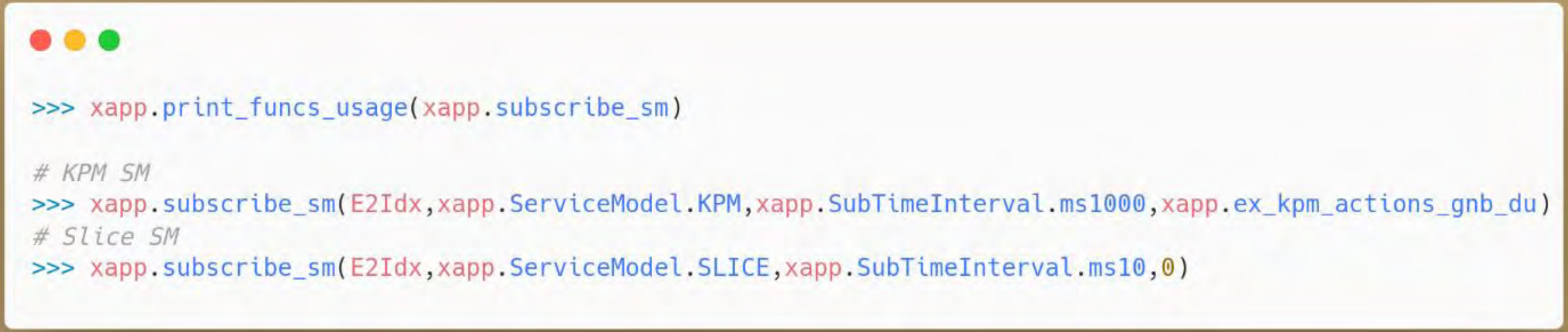
2. Strat xApp



```
>>> xapp.init("path_to_xapp_conf")
```

Containerized xApp
Python xApp with interactive module

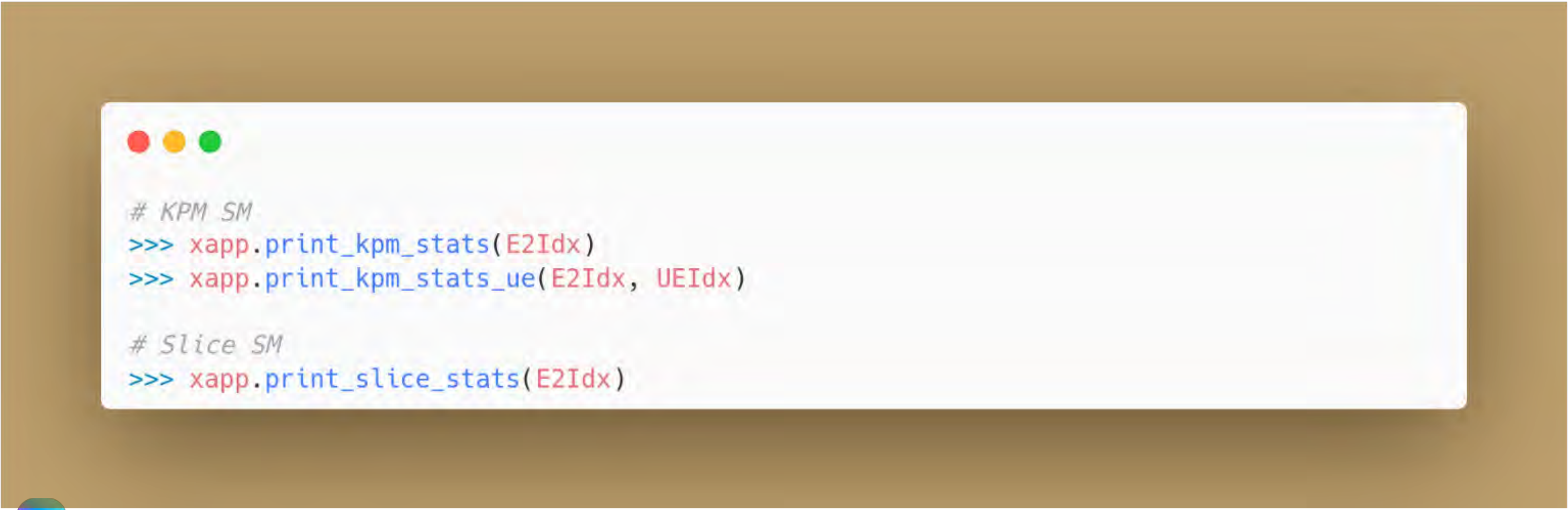
3. Subscribe SM in E2-Node



```
>>> xapp.print_funcs_usage(xapp.subscribe_sm)

# KPM SM
>>> xapp.subscribe_sm(E2Idx,xapp.ServiceModel.KPM,xapp.SubTimeInterval.ms1000,xapp.ex_kpm_actions_gnb_du)
# Slice SM
>>> xapp.subscribe_sm(E2Idx,xapp.ServiceModel.SLICE,xapp.SubTimeInterval.ms10,0)
```

Containerized xApp
Python xApp with interactive module

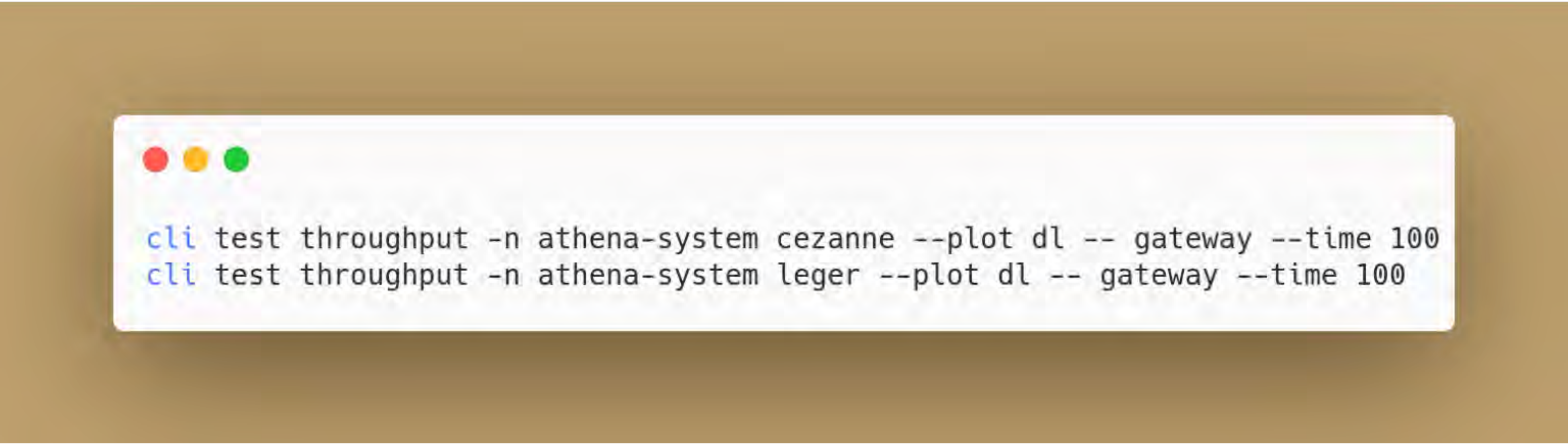
4. Print subscribed SMs' stats



```
# KPM SM
>>> xapp.print_kpm_stats(E2Idx)
>>> xapp.print_kpm_stats_ue(E2Idx, UEIdx)

# Slice SM
>>> xapp.print_slice_stats(E2Idx)
```

Containerized xApp
Python xApp with interactive module

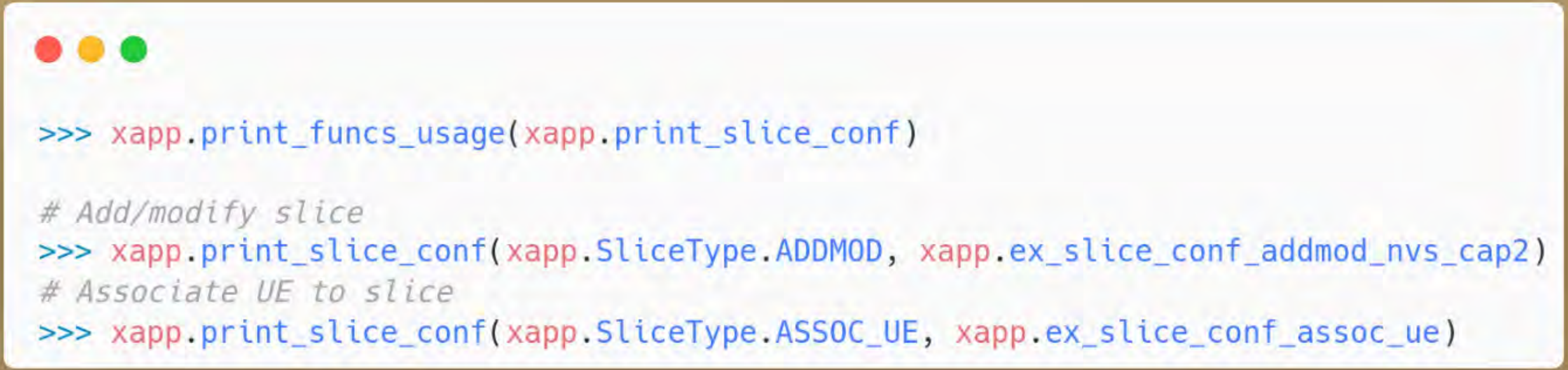
4. Run UEs throughput tests



```
cli test throughput -n athena-system cezanne --plot dl -- gateway --time 100
cli test throughput -n athena-system leger --plot dl -- gateway --time 100
```

Containerized xApp
Python xApp with interactive module

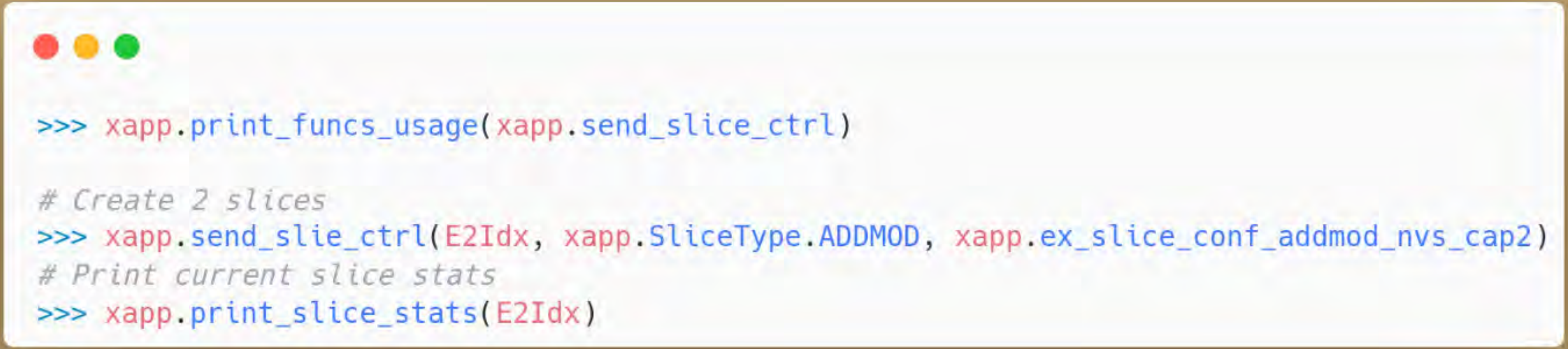
5. Print the slice configuration example



```
>>> xapp.print_funcs_usage(xapp.print_slice_conf)

# Add/modify slice
>>> xapp.print_slice_conf(xapp.SliceType.ADDMOD, xapp.ex_slice_conf_addmod_nvs_cap2)
# Associate UE to slice
>>> xapp.print_slice_conf(xapp.SliceType.ASSOC_UE, xapp.ex_slice_conf_assoc_ue)
```

Containerized xApp
Python xApp with interactive module

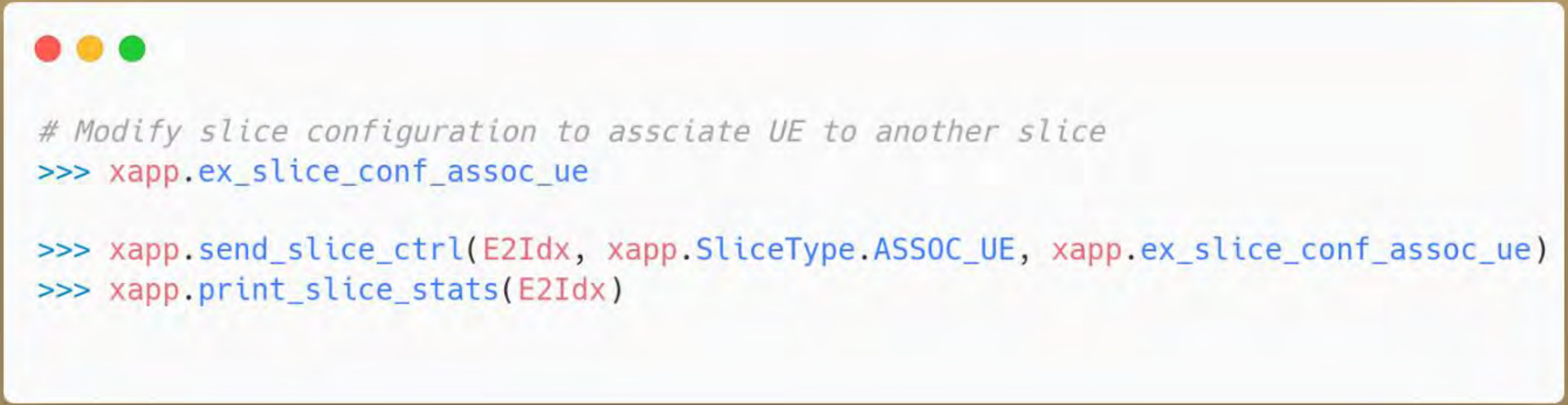
6. Send control message



```
>>> xapp.print_funcs_usage(xapp.send_slice_ctrl)

# Create 2 slices
>>> xapp.send_slie_ctrl(E2Idx, xapp.SliceType.ADDMOD, xapp.ex_slice_conf_addmod_nvs_cap2)
# Print current slice stats
>>> xapp.print_slice_stats(E2Idx)
```

Containerized xApp
Python xApp with interactive module

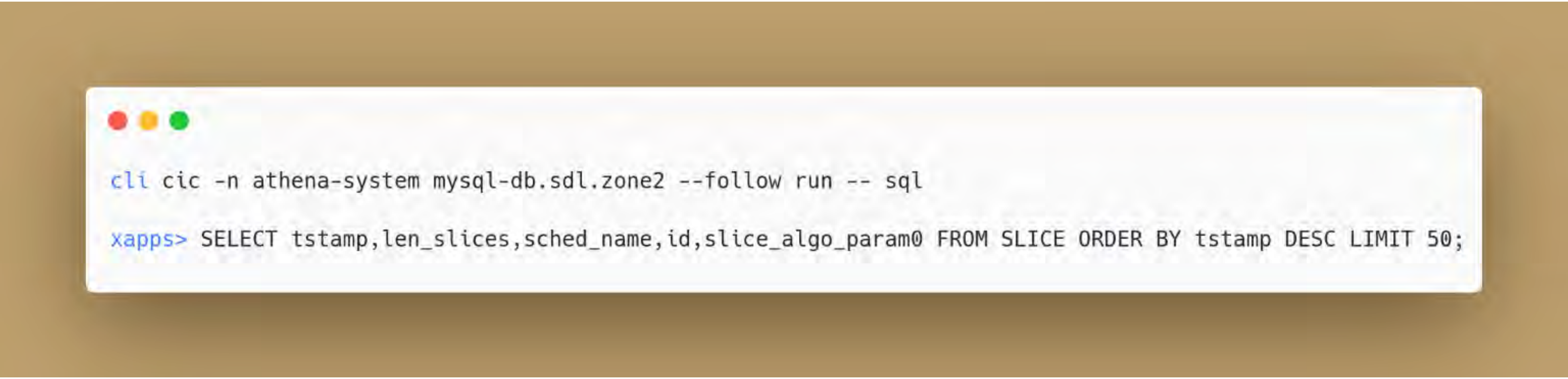
7. Modify the slice configuration and send control message



```
# Modify slice configuration to assciate UE to another slice
>>> xapp.ex_slice_conf_assoc_ue

>>> xapp.send_slice_ctrl(E2Idx, xapp.SliceType.ASSOC_UE, xapp.ex_slice_conf_assoc_ue)
>>> xapp.print_slice_stats(E2Idx)
```

Containerized xApp
Python xApp with interactive module

8. Open MySQL database to query the collected Slice data

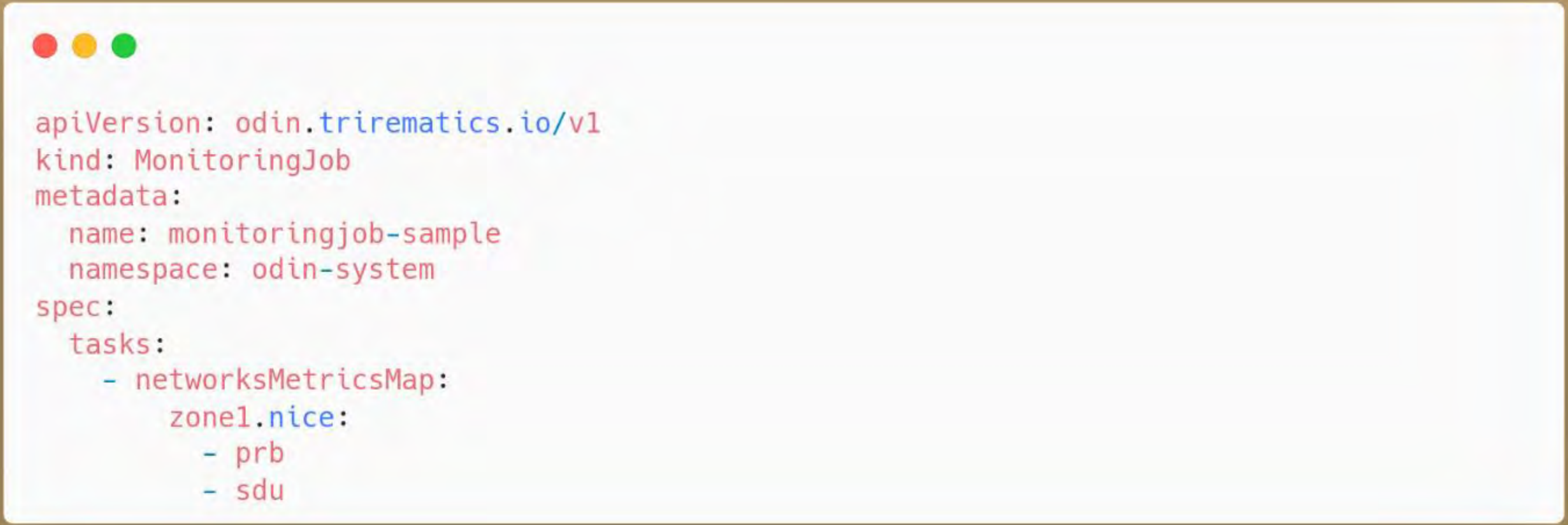


```
cli cic -n athena-system mysql-db.sdl.zone2 --follow run -- sql

xapps> SELECT tstamp,len_slices,sched_name,id,slice_algo_param0 FROM SLICE ORDER BY tstamp DESC LIMIT 50;
```

	xApp DevOps		Data & ML	
Open RAN Ecosystem	Binary xApp	Containerized xApp	Integrated xApp	rApp
Roles	**Developer**	**Maintainer**	**Vertical User**	**Business intelligent & Data analyst**
Participants	**Vendor**	**Operator**	**Application Provider**	**Stakeholder**
Usage Scenario	**Research & Develop**	**Testing & Measurement**	**Production**	**Analysis & Optimize**
Knowledge of NearRT-RIC APIs	**Proficient**	**Moderate**	**Basic**	**None**
Knowledge of SMO APIs	**None**	**Basic**	**Moderate**	**Proficient**
Actions	**Develop new functionalities**	**Interact with network**	**Apply & Enforce policy**	**Define network intent & Create policy**
Programming Languages	C/C++, Python, Go	Python, Go	Python, Go	Any
Network Configuration	Manual	Automatic	Automatic	Automatic
Network Deployment	Static	Static	Dynamic	Dynamic
Demo Example	C xApp with monitor service	Python & Go xApp with interactive module	Go xApp with A1 APIs & Programmable Python xApp	Coming Soon

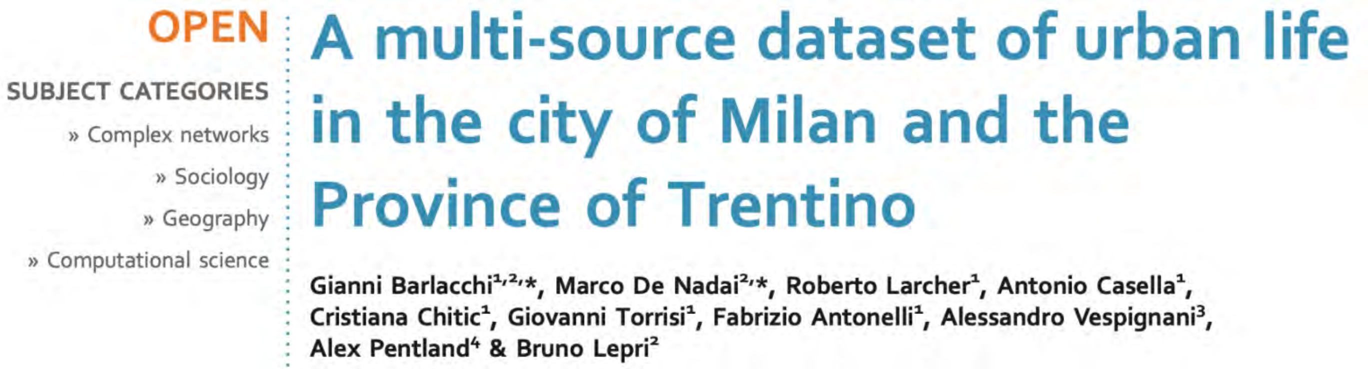
Integrated xApp
Programmable Monitoring

9. Deploy a monitoring job request to ODIN



```yaml
apiVersion: odin.trirematics.io/v1
kind: MonitoringJob
metadata:
  name: monitoringjob-sample
  namespace: odin-system
spec:
  tasks:
    - networksMetricsMap:
        zone1.nice:
          - prb
          - sdu
```

Programmable Monitoring - A need that starts 2020



OPEN

SUBJECT CATEGORIES

» Complex networks

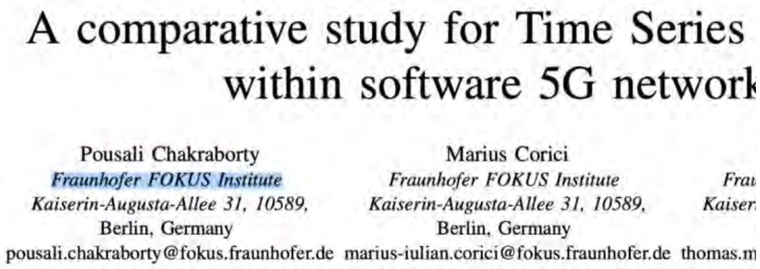
» Sociology

» Geography

» Computational science

A multi-source dataset of urban life in the city of Milan and the Province of Trentino

Gianni Barlacchi[1,2,*], Marco De Nadai[2,*], Roberto Larcher[1], Antonio Casella[1], Cristiana Chitic[1], Giovanni Torrisi[1], Fabrizio Antonelli[1], Alessandro Vespignani[3], Alex Pentland[4] & Bruno Lepri[2]

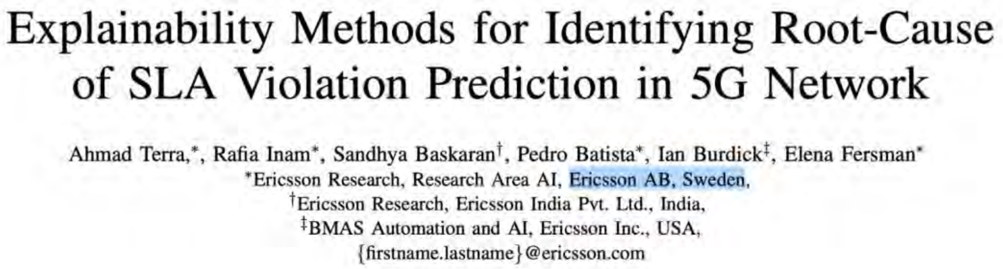


A comparative study for Time Series within software 5G network

Pousali Chakraborty
Fraunhofer FOKUS Institute
Kaiserin-Augusta-Allee 31, 10589,
Berlin, Germany
pousali.chakraborty@fokus.fraunhofer.de

Marius Corici
Fraunhofer FOKUS Institute
Kaiserin-Augusta-Allee 31, 10589,
Berlin, Germany
marius-iulian.corici@fokus.fraunhofer.de

Frau
Kaiser
thomas.m



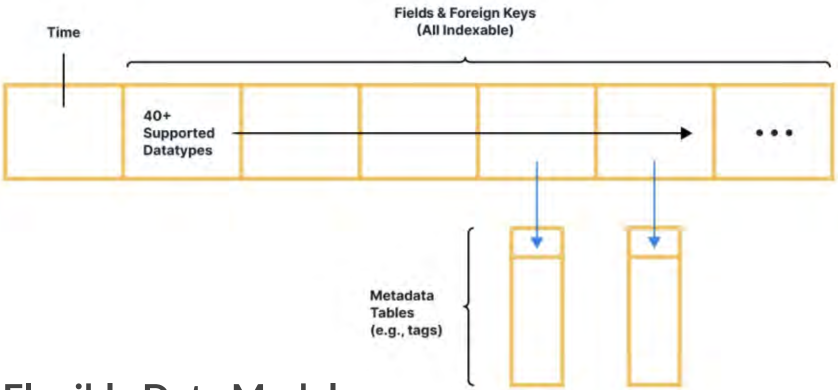
Explainability Methods for Identifying Root-Cause of SLA Violation Prediction in 5G Network

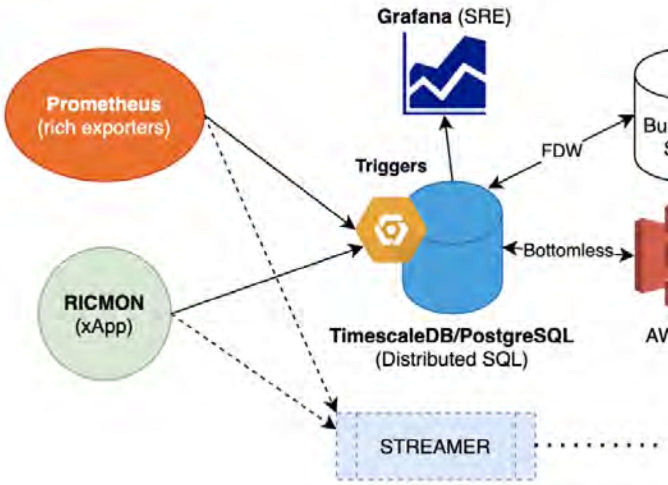
Ahmad Terra,[*], Rafia Inam[*], Sandhya Baskaran[†], Pedro Batista[*], Ian Burdick[‡], Elena Fersman[*]
[*]Ericsson Research, Research Area AI, Ericsson AB, Sweden,
[†]Ericsson Research, Ericsson India Pvt. Ltd., India,
[‡]BMAS Automation and AI, Ericsson Inc., USA,
{firstname.lastname}@ericsson.com

- ★ Even for the same scenario, optimal learner of A **does not** translate to optimal of A’.
- ★ We couldn't collect data on the “real” testbed like big labs, stick to this 2011 dataset…

FlexMon: Goals & Architecture

1) Get ALL the data you want, **plus programmability**, in the simplest way.
2) Worry-free about Multi-source Schema & Scaling: Reliability + FDW + S3.
3) Beyond Jupyter Notebooks into Production: SRE + Streamer.





Flexible Data Model:

- *Ingestion w/o knowing the schema.*
- *Preprocess with few assumptions.*

FlexMon - DEMO

- ★ Programmable Monitoring as simple as Copy-and-Paste.
- ★ **Lake, act 1**: PromQL for EZ Visualization (network monitoring).
- ★ **Lake, act 2**: SQL for Precise Data Analysis (machine learning).

- ★ *Will Python dynamicity affect our performance?* Testing FlexMon at Scale.

  Hint: 4x Wikipedia at ½ Latency.

Today's Agenda and Speakers

Part 1 (45 minutes)

1. Open RAN Studio: Features and Bronze Release Notes
2. Non-RT RIC: Architecture and rApps call flow
3. OAM: How to design and deploy a 5G Open RAN network on GKE

Break (5 minutes)

Part 2 (45 minutes)

1. xApp lifecycle: RAN slicing use-cases
2. Data Analytics: Large-scale 5G Open RAN deployment
3. DevOps xApp: Interactive xApp
4. Observability: Data flow processing
5. Guest Demo (10 minutes)
   a. Interoperability between Open RAN Studio and OSC DU
6. Closing remarks and Q&A (10 minutes)

Alireza
BubbleRAN
Product Manager

Ilias
Eurecom
SMO expert

Chieh-Chun
Eurecom
RIC expert

Khoa
Eurecom
Data Scientist

Ian
NTUST-BMWLab
PhD student

© BubbleRAN

Closing Remarks

- **Open RAN Mini-Series are a forum to share knowledge and foster academia-industry collaboration**
  - We value your feedback to improve the quality, format and the content of upcoming series

- **To this end, Open RAN Studio platform is designed with the following objectives**
  1. Empower communities and organizations to accelerate the adoption of modern technologies
  2. Solid ground for tutoring the next generation researchers and engineers
  3. Reproducible/verifiable and consistent outcomes for teaching and research
  4. Affordable and accessible means for education and research
  5. Opening new possibilities and dimensions via multi-disciplinary research

Linkedin: https://www.linkedin.com/company/bubbleran
Youtube: https://www.youtube.com/@bubbleran

Feedback

https://bubbleran.com/products/mx-ors/

Q & A

“

© BubbleRAN