www.6g-leader.eu

LEARNING-DRIVEN AND EVOLVED RADIO FOR 6G COMMUNICATION SYSTEMS

Funded by the European Union

6GSNS

Funded by the European Union. The project is supported by Smart Networks and Services Joint Undertaking (SNS JU) and its members. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or SNS JU. Neither the European Union nor the granting authority can be held responsible for them.

www.6g-leader.eu

RIC and xApp development

The 6G-LEADER vision

Dr. German Castellanos
Dr. Ensar Zeljkovic
Accelleran

Funded by the European Union

6GSNS

Funded by the European Union. The project is supported by Smart Networks and Services Joint Undertaking (SNS JU) and its members. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or SNS JU. Neither the European Union nor the granting authority can be held responsible for them.

Agenda

1) Introduction to 6G-LEADER O-RAN-based Architecture and extension

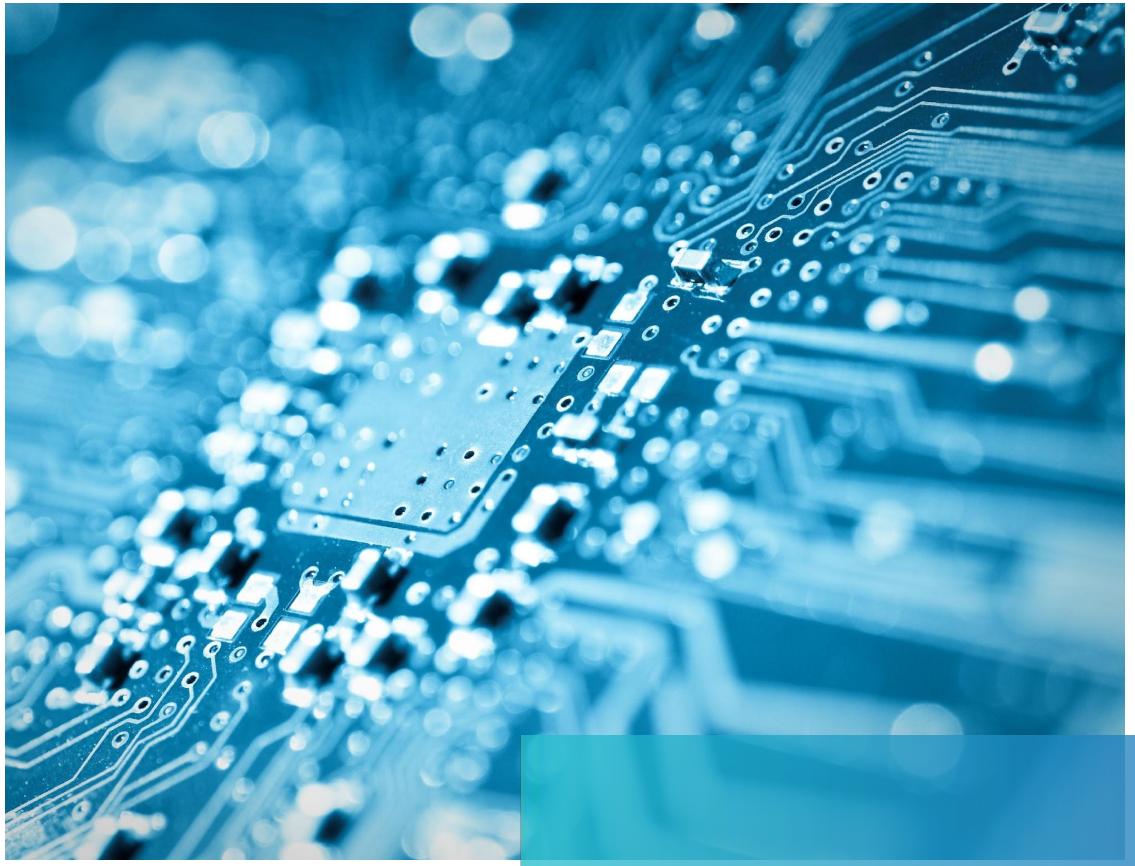
2) dRAX as a RIC & SMO solution

5 min 

coffee break

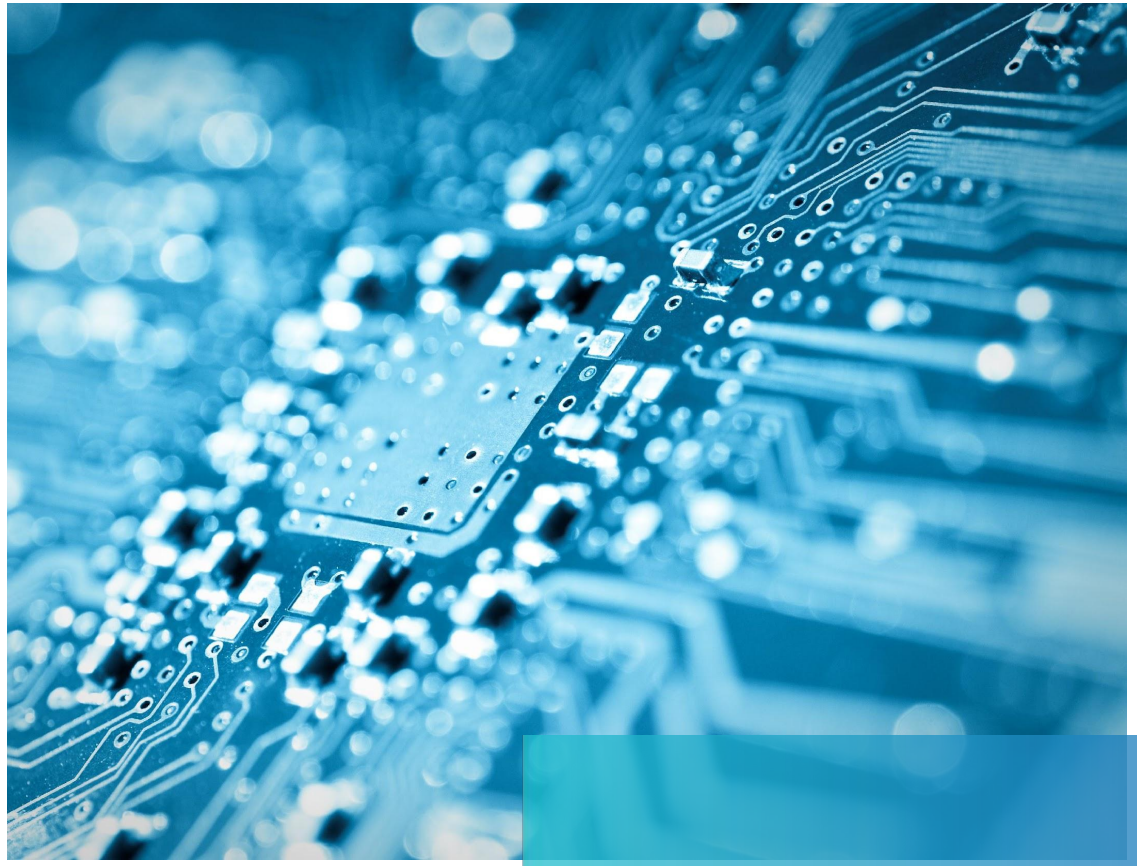
3) x/rApp Live Development

4) Conclusions and Q&A

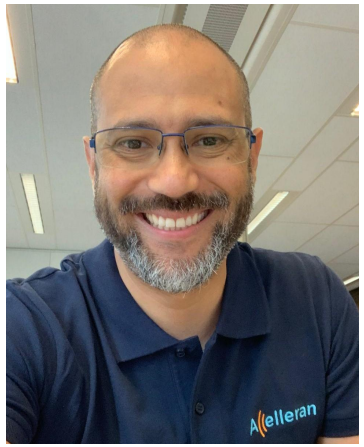


Agenda

1) Introduction to 6G-LEADER O-RAN-based Architecture and extension (German) 0:10
2) dRAX as a RIC & SMO solution (Ensar) 0:45
    - 2.1) Introduction
    - 2.2) dRAX Architecture & Features
    - 2.3) x/rApp Development Framework
    - 2.3) KPMs and RAN Control
    - 2.4) x/rApp Development Environment
3) x/rApp Live Development (Ensar) 1:45
    - 3.1) KPM/PM Monitoring
    - 3.2) Data Enrichment & Inter-x/rApp Comm.
    - 3.3) dRAX Accelleran RAN Events Monitoring
    - 3.4) RAN Discovery and configuration
    - 3.5) x/rApp Discovery and configuration
    - 3.6) x/rApp API
    - 3.7) dRAX Notifications & Alarms
4) Conclusions and Q&A (German & Ensar) 0:20



Presenters



Driving the **future of private and open 5G** with O-RAN architectures and RIC innovation.

Leading **European projects** on intelligent RAN orchestration, with xApps for energy savings, traffic steering, and safe conflict management.

Shaping **AI-native, scalable RIC platforms** that pave the way to 6G.

Dr. German Castellanos - RAN System Architect at Accelleran

Leading the **product roadmap** for Accelleran's OpenRAN-inspired RIC in private 5G.

Driving innovation with a **low-code SDK** that accelerates xApp/rApp development and customer adoption.

Bridging **deep tech and business strategy** to fuel product growth and industry impact.

Dr. Ensar Zeljkovic - RIC Product Owner at Accelleran

The project

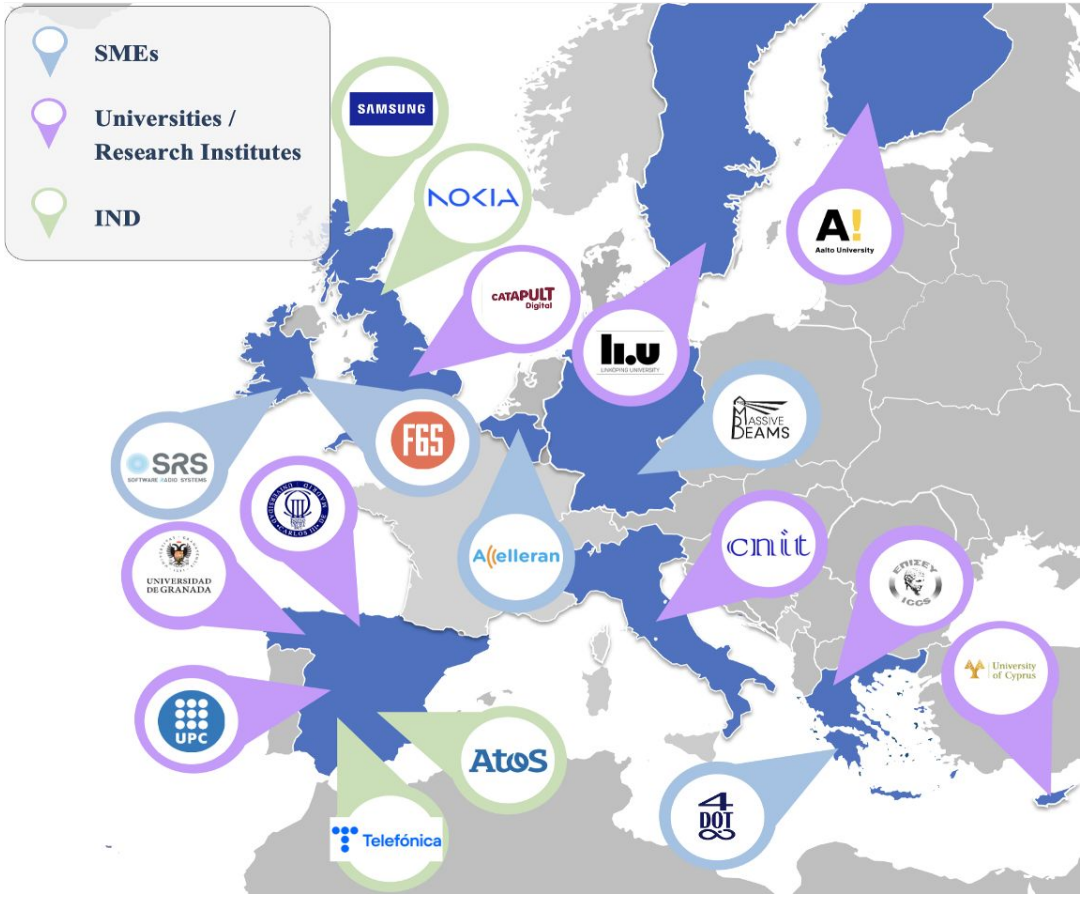
6G-LEADER at a glance

CALL TOPIC

HORIZON-JU-SNS-2024-STREAM-B-01-02

DURATION

36 MONTHS
1 JAN 2025 - 31 DEC 2027



10 COUNTRIES
18 PARTNERS

6G-LEADER in a nutshell

6G-LEADER advances the design of future wireless systems by **integrating AI/ML**, **semantic communications**, and **reconfigurable RF technologies** into Open RAN.

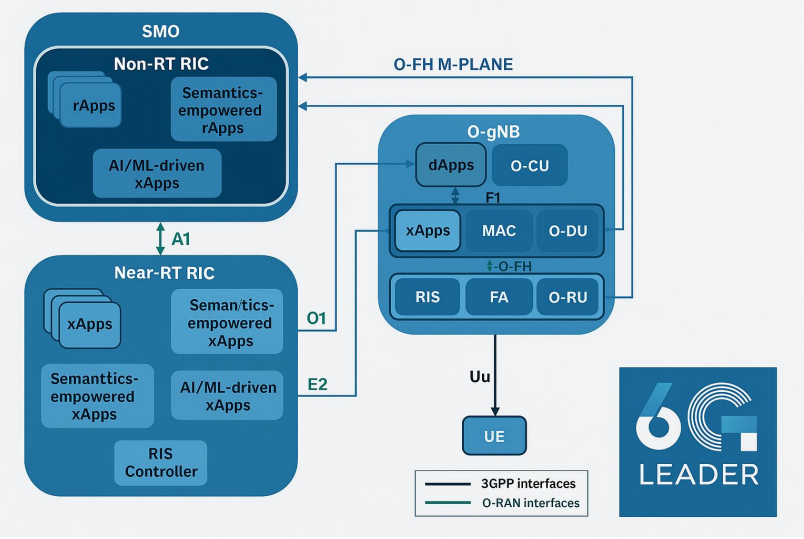
The project develops innovative RAN control through rApps, xApps, and dApps, targeting:

- Real-time and near-real-time intelligent network management
- Enhanced energy efficiency and reduced EMF exposure
- Conflict mitigation between applications for robust RAN operation
- Proof-of-Concepts validating 6G innovations in real testbeds

RIC and xApps: The Core of the 6G-LEADER Architecture

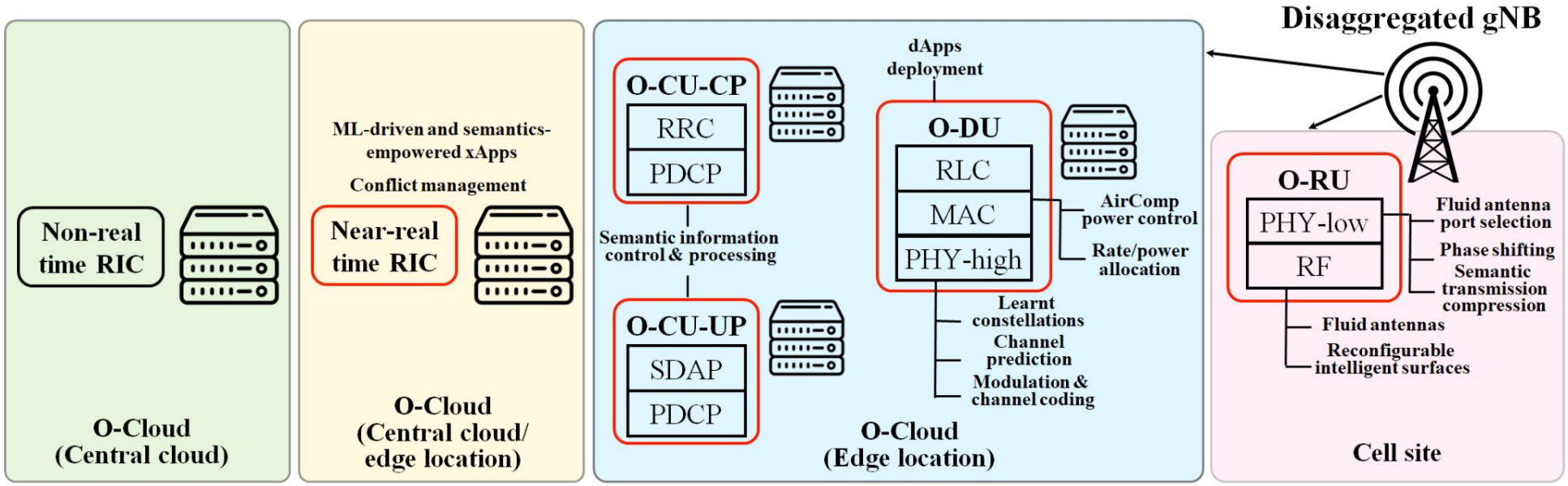
At the heart of **6G-LEADER** is a layered O-RAN architecture where the RIC orchestrates intelligent control:

- **Non-RT RIC (rApps)**: high-level policy and long term optimisation
- **Near-RT RIC (xApps)**: fast, AI/ML-driven decisions with semantic awareness
- **dApps**: ultra-low latency distributed control loops (<10 ms) at DU/RU level
- **Conflict Manager**: ensures harmony among multiple xApps and dApps, safeguarding QoS and reliability



6G-LEADER architecture

- **6G-LEADER** adopts O-RAN's **disaggregation framework**, dividing the gNB into O-CU, O-DU, and O-RU and developing solutions for:
  - RAN control and optimisation
  - ML-driven and semantics-empowered wireless communication technologies
  - Signal processing algorithms and innovations on FAs and RISs



6G-LEADER architecture

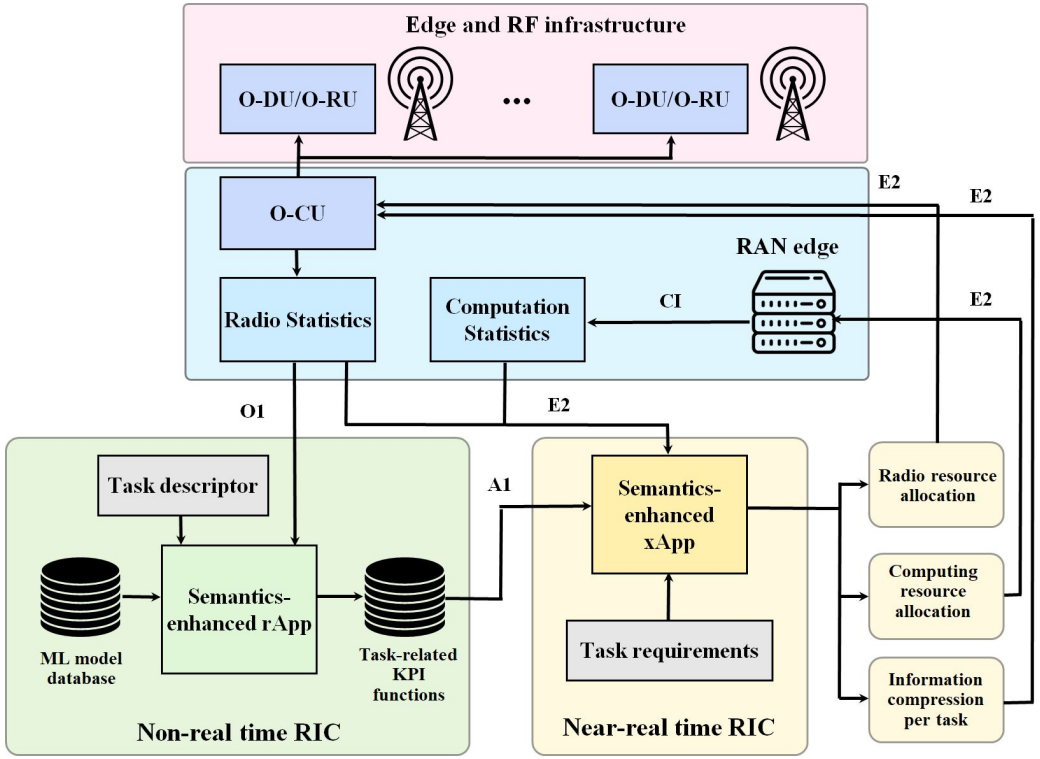
- **6G-LEADER**'s architecture includes closed-loop RAN control, based on RIC
- Will enhance the **two O-RAN closed control loops**, operating in non-real-time (**rApps**-based) and near real-time (**xApps**-based) with AI/ML techniques and semantic awareness.
- **Real-time RAN control** via distributed Apps (**dApps**), supporting a **third closed control loop** between the O-DU and O-RU in less than 10 ms
- The project will introduce a **Conflict Manager** within the near-RT RIC solving conflicts between the ML-driven and semantics-empowered xApps and dApps

6G-LEADER architecture (Semantics)

- **Semantics integration in O-RAN** enhancing dApps, xApps, rApps

- **Benefits**:
  - Reduced communication overheads.
  - Enhanced multi-access edge computing.

- Semantics-aware tasks consist of **Task Descriptor (TD)** and **Task Requirements (TR)** fields

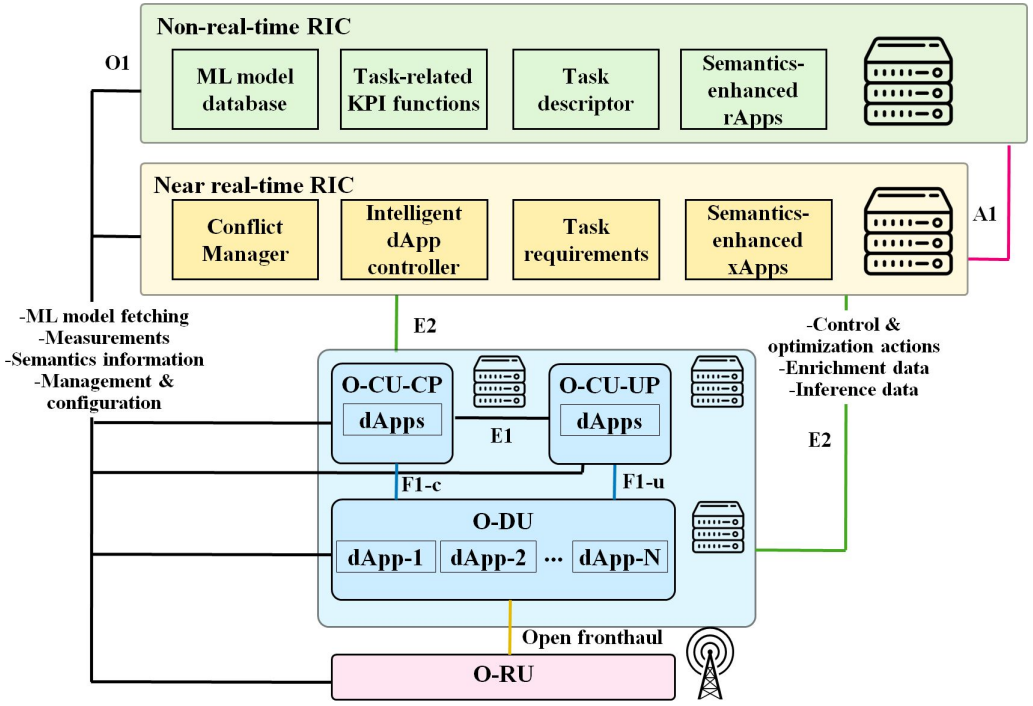


6G-LEADER architecture (dApps)

- 6G-LEADER will offer **dApps**, operating at sub-10 ms timescales

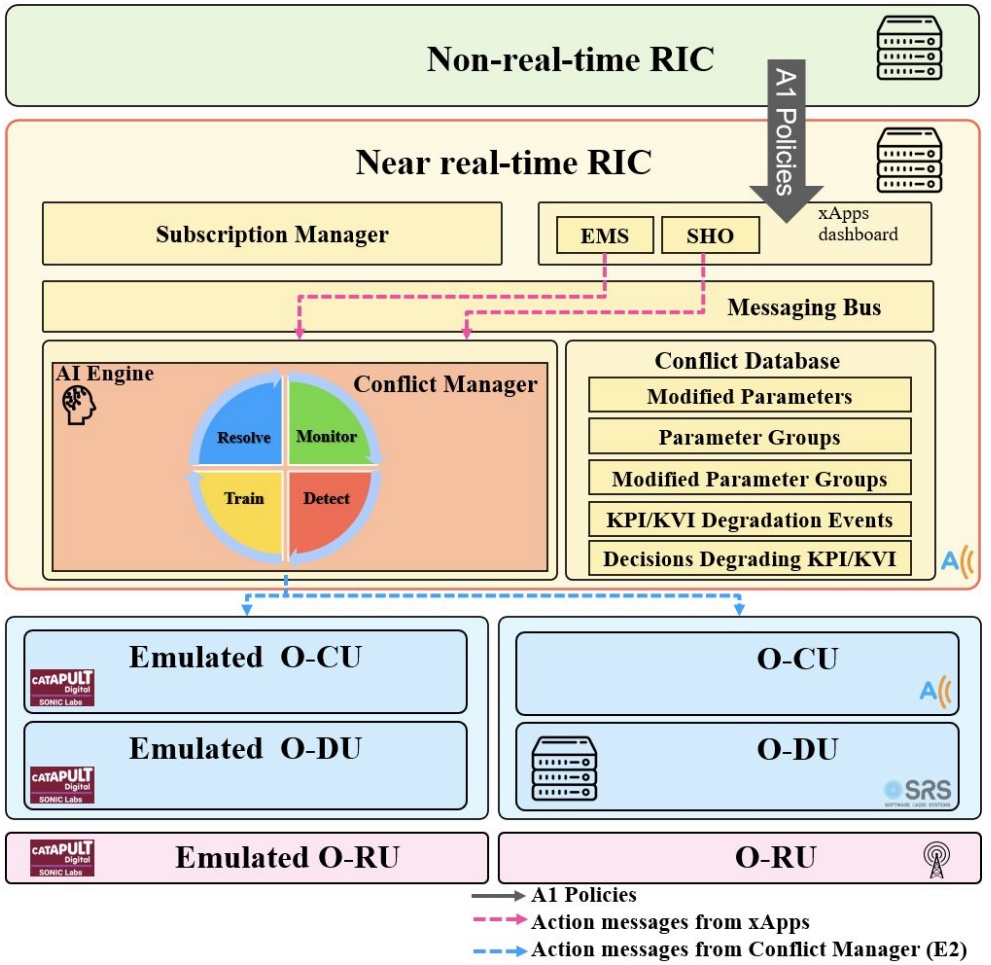
- **Benefit**:
  - Real-time inference
  - Supporting RAN functionalities beyond current O-RAN architectures

- Similar to xApps and rApps, dApps wil adopt a containerized architecture for:
  - Managing the dApps life cycle.
  - Simplifying the integration.
  - Establishing an abstraction level enabling dApps to control RAN parameters
  - Implementing H/W-independent dApps and standalone O-RAN applications



6G-LEADER architecture (Conflict Manager)

- 6G-LEADER includes a **Conflict Manager**, operating across the different control and optimisation loops
- **Benefit**:
  - The impact of direct, indirect and implicit conflicts is mitigated
- For xApps, this component handles potential conflicts as xApps may implement conflicting configurations at different optimisation goals
- Deploying dApps requires conflict mitigation to identify conflicts among xApps, and dApps



Who is Aelleran?

An **API-First** management approach creates an ultimately open and integratable solution.
Pre-integrated with broad range of **Cores**.

Easy-to-Integrate

The **Open Fronthaul Interface** brings in ORUs from across the O-RAN ecosystem.

Easy-to-source

Accelleran

A telecom infrastructure software company

Delivering turn-key solutions for 5G private networks

A **fully virtualized 5G RAN solution**, containerized, K8S deployed, cloud-native, running uniquely on **Commercial-off-the-Shelf** hardware, delivering the full promise of 5G

Easy-to-Deploy and Maintain

The integrated **RAN Intelligent Controller** with a well-documented **Software Development Kit**, allows easy optimisation and automation for specific use cases.

Easy-to-Optimise

Accelleran Private 5G solution components

Accelleran Dashboard

3rd party NMS

Private 5G REST API

Service Management & Orchestration (SMO)

Accelleran

Accelleran RIC

Accelleran DU

Accelleran CU

5G Radio Units

O-RAN FH CUS-Plane & M-Plane

3GPP N2/N3

5G Core

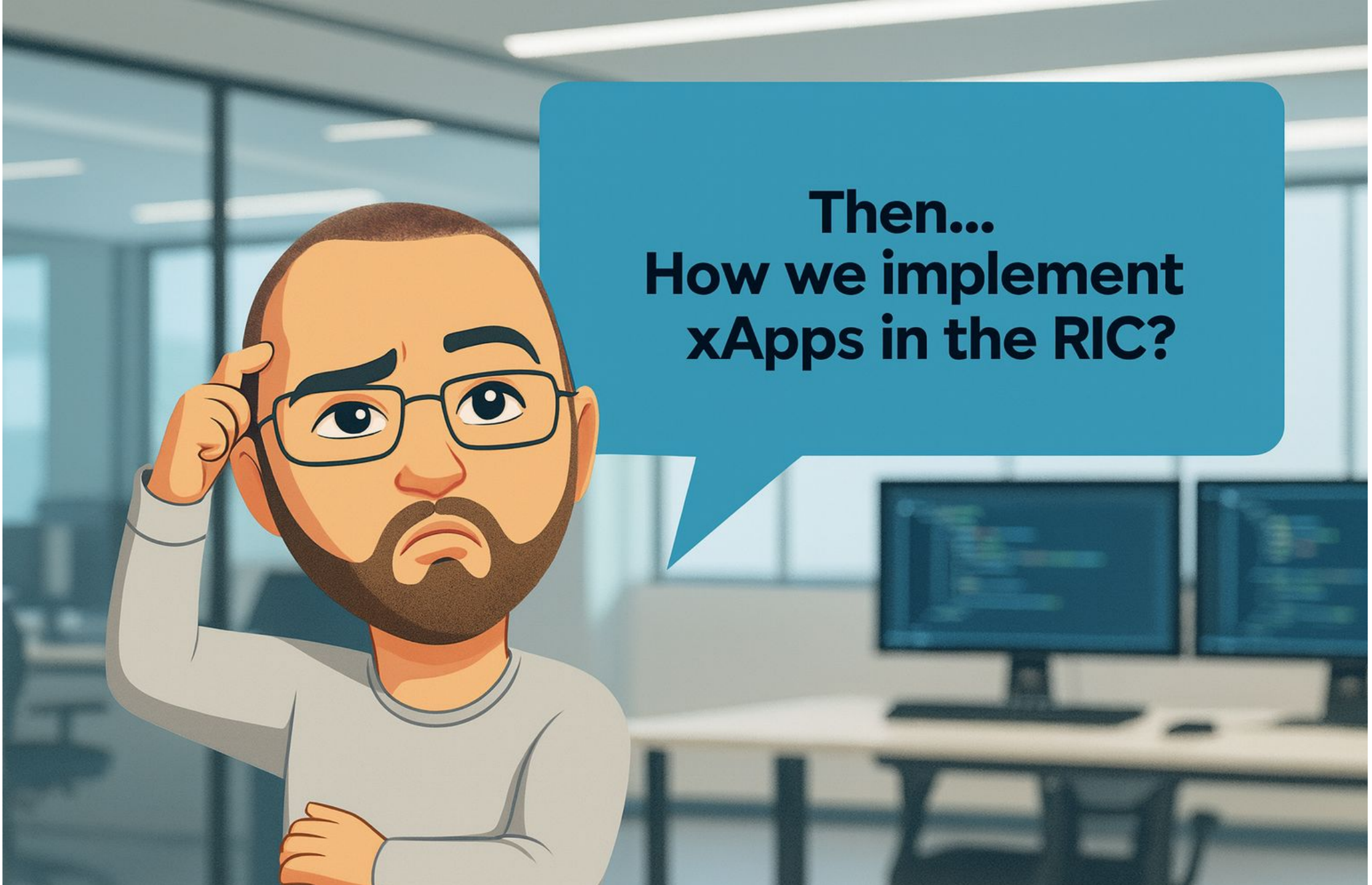
Open5GS
MAVENIR
Hewlett Packard Enterprise
Druid
ATTOCORE
Cumucore
KEYSIGHT
metaswitch

Accelleran Software

COTS Servers

Accelleran Industry Focused 5G SA Solution

RIC


Then...
How we implement
xApps in the RIC?

RIC
Radio Intelligent Controller

Why do we need a RIC?

What if your network could optimize itself in real-time, save you money, and open up new services – all while using open standards?

The challenge: Modern networks are highly complex and static configurations can’t keep up with dynamic demands.

Introducing the RIC – the brain of the Open RAN that brings adaptability and intelligence

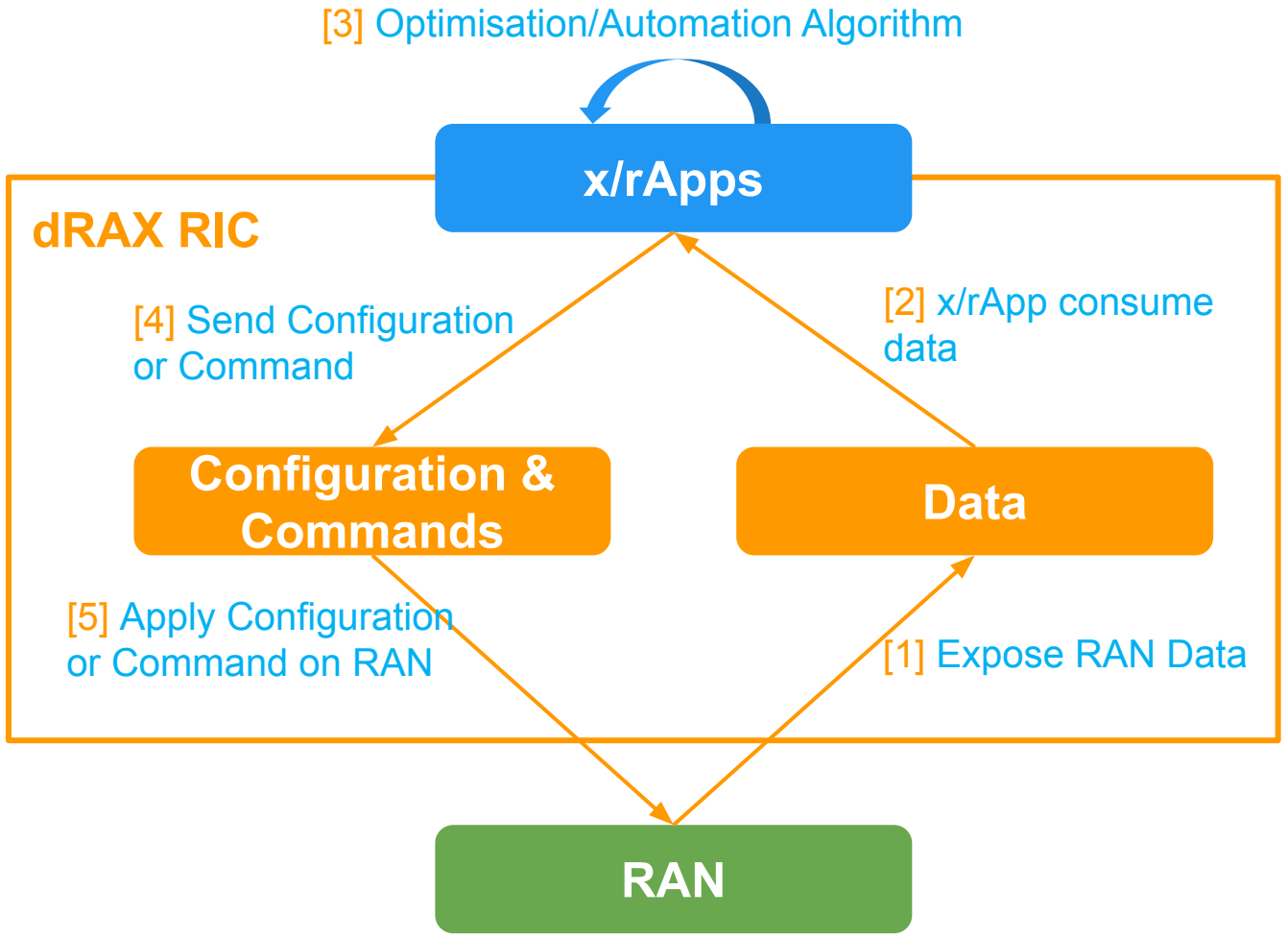
Key Features and Benefits

Enables Real-Time, Intelligent RAN Control: Supports **diverse** use cases by providing the agility and responsiveness needed to meet complex network demands across **multiple verticals**.

Programmable and Extensible Platform: Offers a flexible environment for deploying **xApps and rApps**, allowing operators to **dynamically optimize and automate** RAN functions.

Data-Driven Insights and Precision Control: Leverages a wide array of **KPMs and PMs** from 5G CU and DU, providing operators with in-depth visibility into **network performance and health**.

Fine-Tuned Network Adjustments: Equipped with **control hooks** for precise network behavior modifications, empowering use cases like **interference management, admission control, and traffic steering**.



Key Use Cases for the RIC and x/rApps

Interference Management: xApps can dynamically **adjust transmission power** or initiate **handovers** to mitigate interference between cells, especially in dense urban areas or during high traffic events.

Traffic Steering: With near-real-time insights into **network load**, xApps can **balance traffic** between cells or steer users to the optimal cell, enhancing user experience and **optimizing resource utilization**.

Energy Efficiency: x/rApps can be developed to intelligently manage network energy consumption, such as **turning off underutilized cells** or **optimizing power** based on real-time demand.

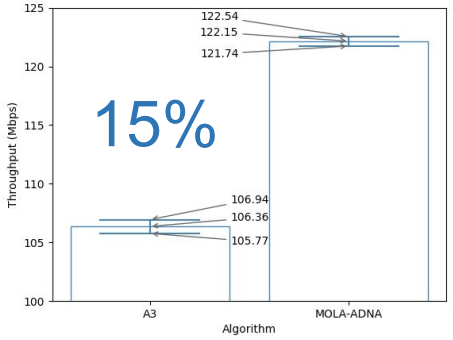
Handover Optimization: The RIC, through xApps, can **fine-tune handover** parameters to minimize service interruptions and ensure a **seamless experience** for users moving across cells.

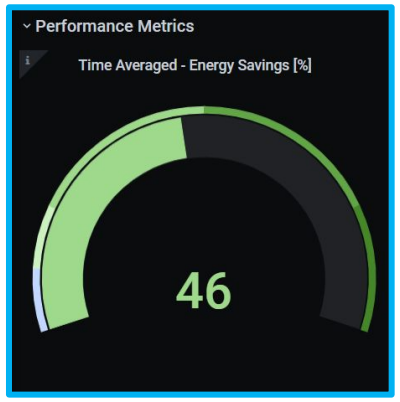
Anomaly Detection and Predictive Maintenance: x/rApps can continuously **monitor network KPIs** to detect **anomalies**, identify potential failures, and **trigger corrective actions** before network performance is impacted.

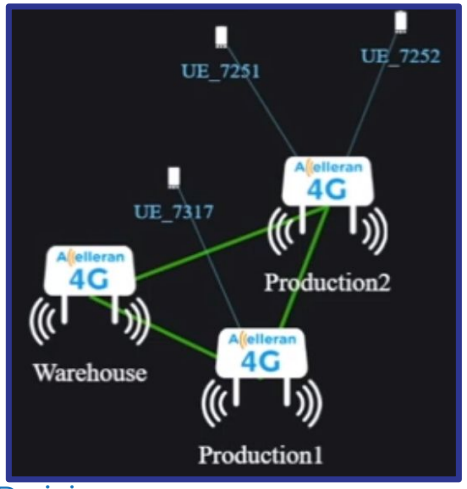
Admission Control: x/rApps can **dynamically manage the admission** of new users or services into the network, based on real-time resource availability and network conditions.

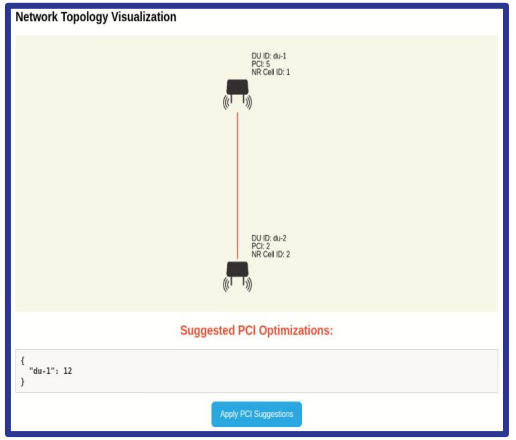
And many more!

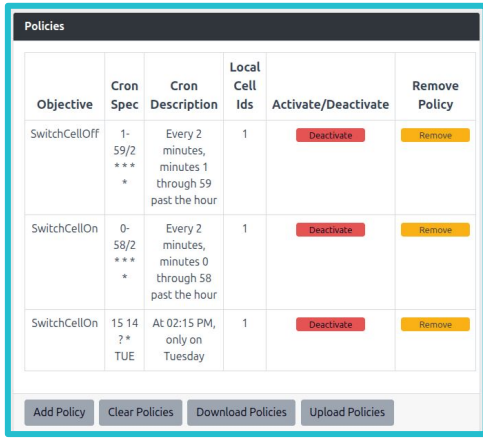
Accelleran x/rApps & Use Cases





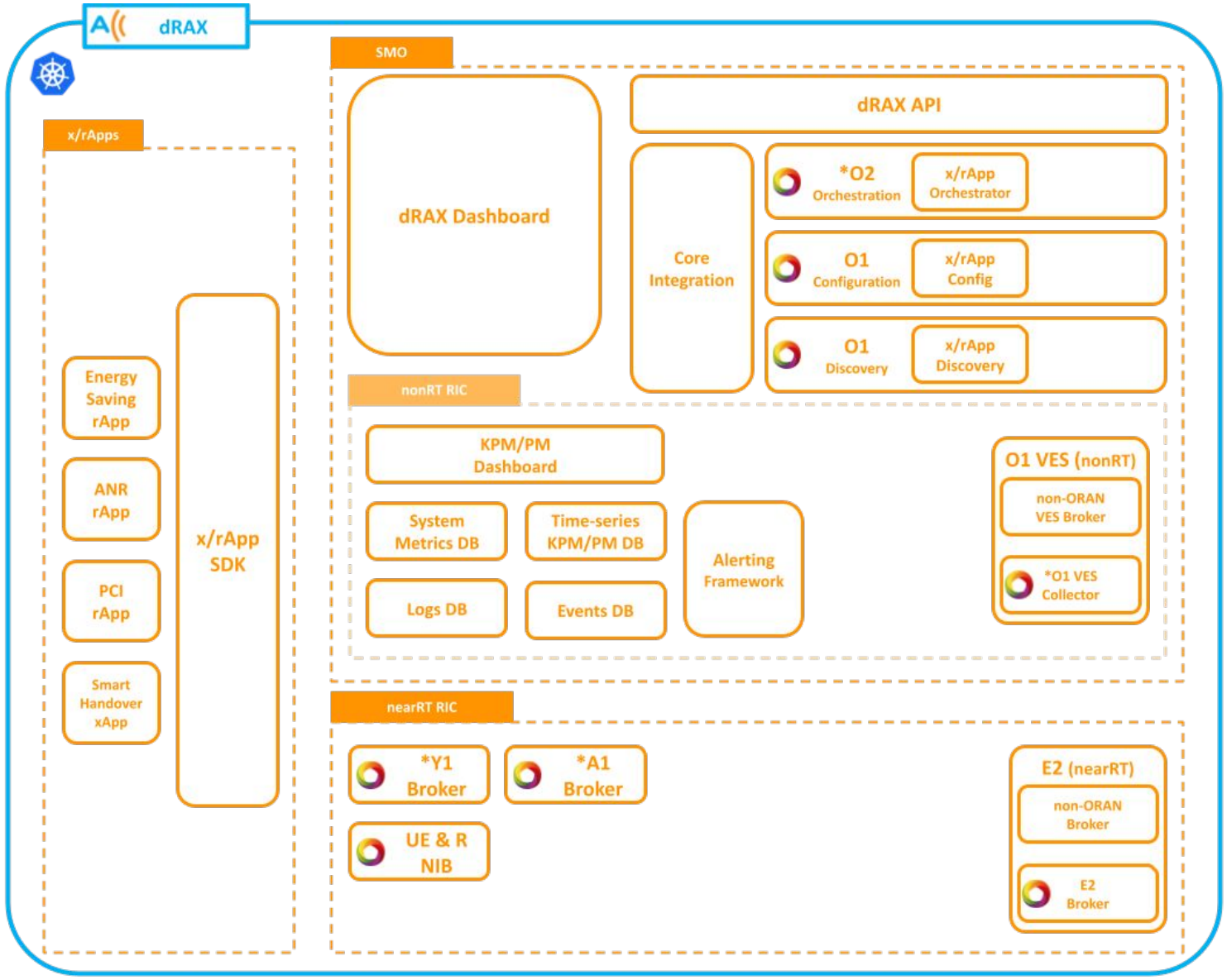






Accelleran dRAX RIC

dRAX Functional Architecture



Microservice oriented approach

Combined platform taking advantage of Kubernetes flexibility

nearRT and nonRT RIC created on the same platform

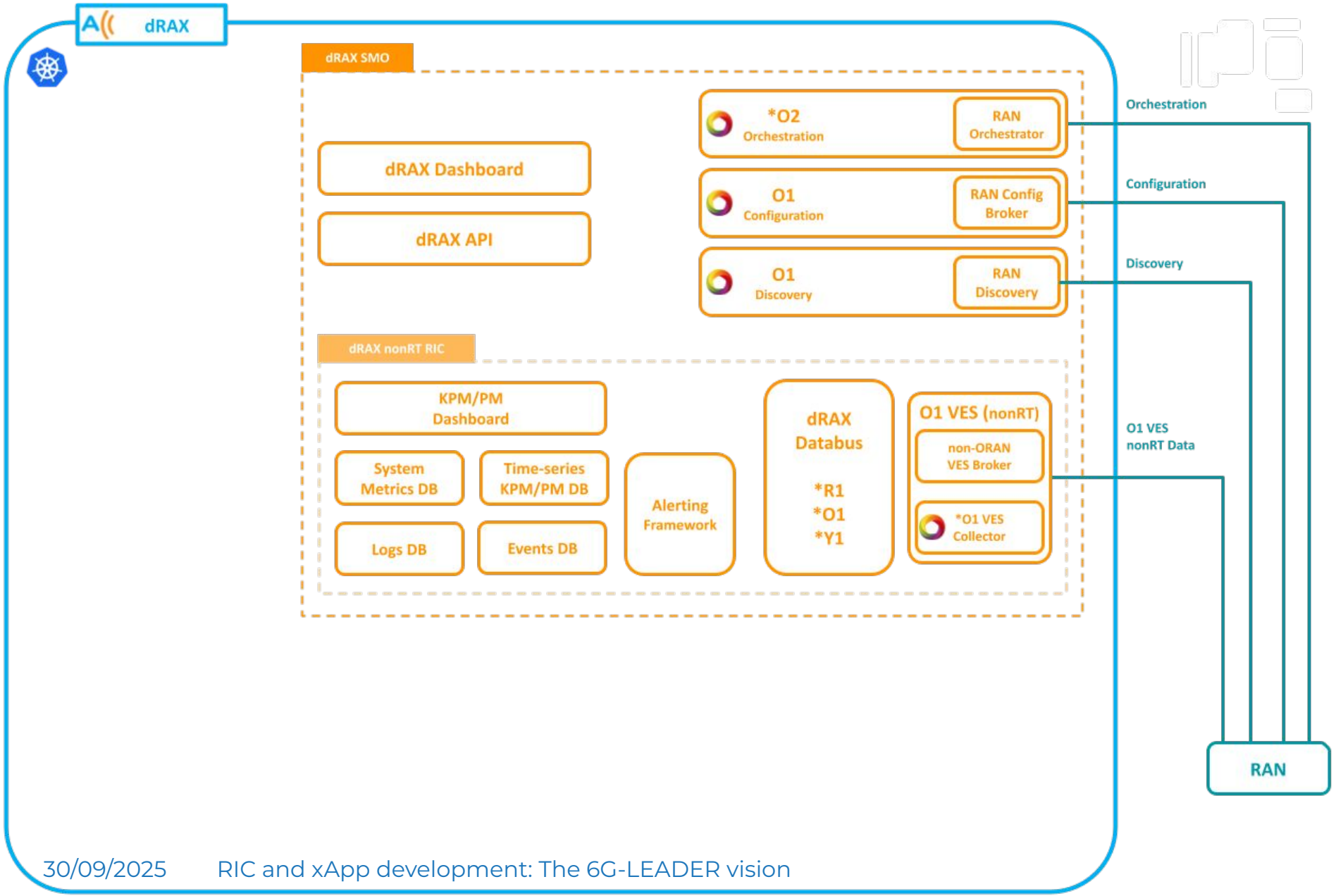
Unified rApp & xApp development experience with one SDK

Differentiate based on use case

Ability to create xApps, rApps but also hybrid Apps taking advantage of both nearRT and nonRT RIC functionality

xApp - rApp communication is a given

dRAX SMO - RAN



dRAX Dashboard:
dRAX GUI for easy usability

dRAX API:
API gateway to all dRAX APIs exposed

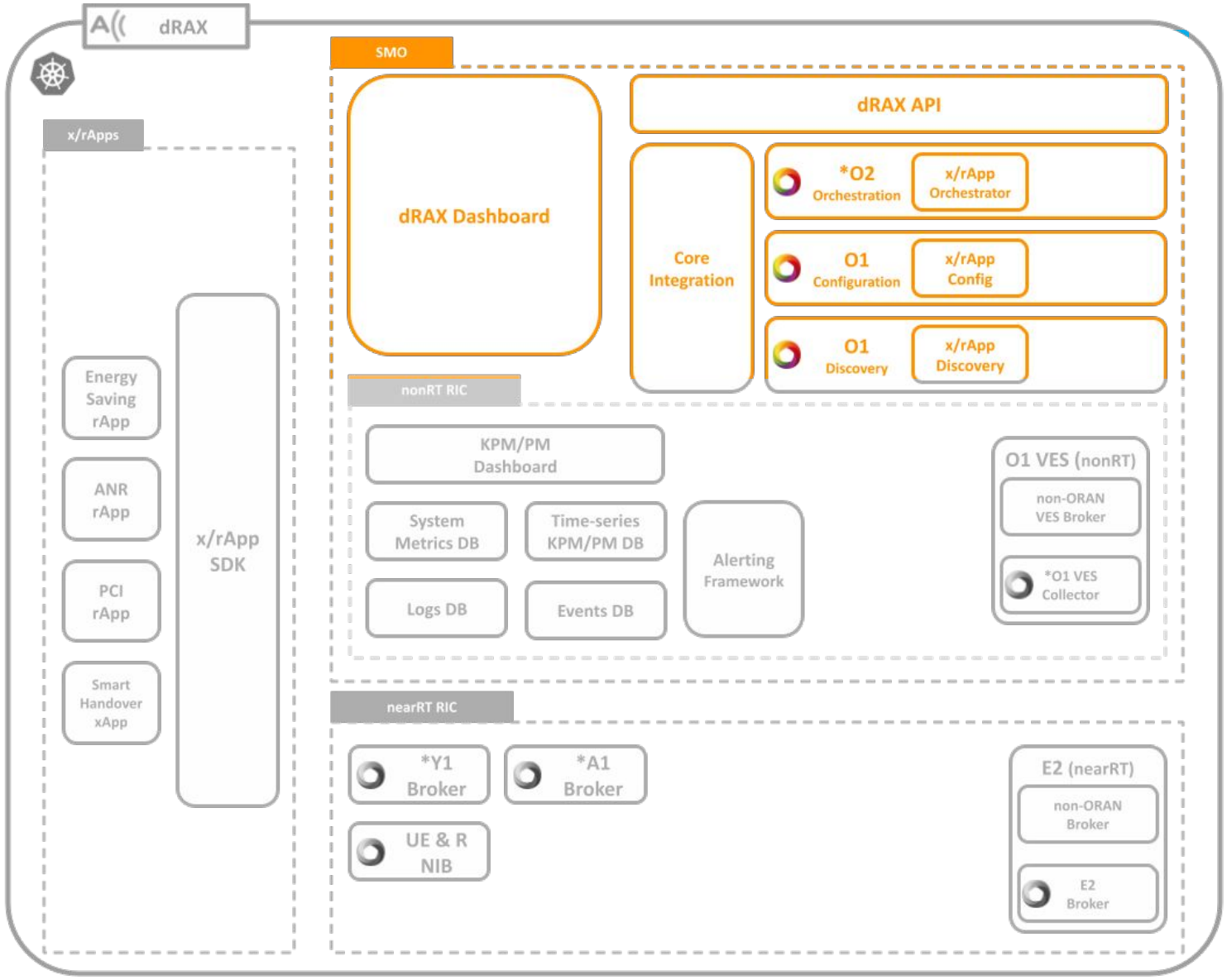
O1 Orchestration:
Onboarding & lifecycle management of RAN components

O1 Configuration:
Configuration of RAN components

O1 Discovery:
Discovery of RAN components

O1 Abstraction:
Abstractions for non-ORAN aligned RAN vendors

dRAX SMO - x/rApps



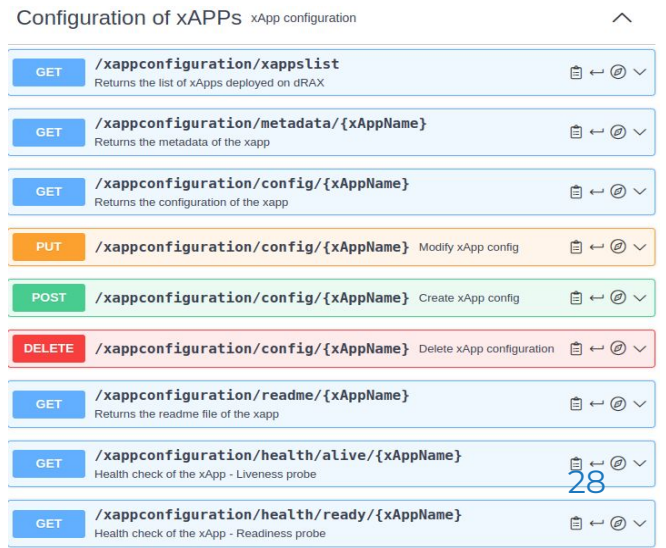
dRAX Dashboard:
dRAX GUI for easy usability

dRAX API:
API gateway to all dRAX APIs exposed

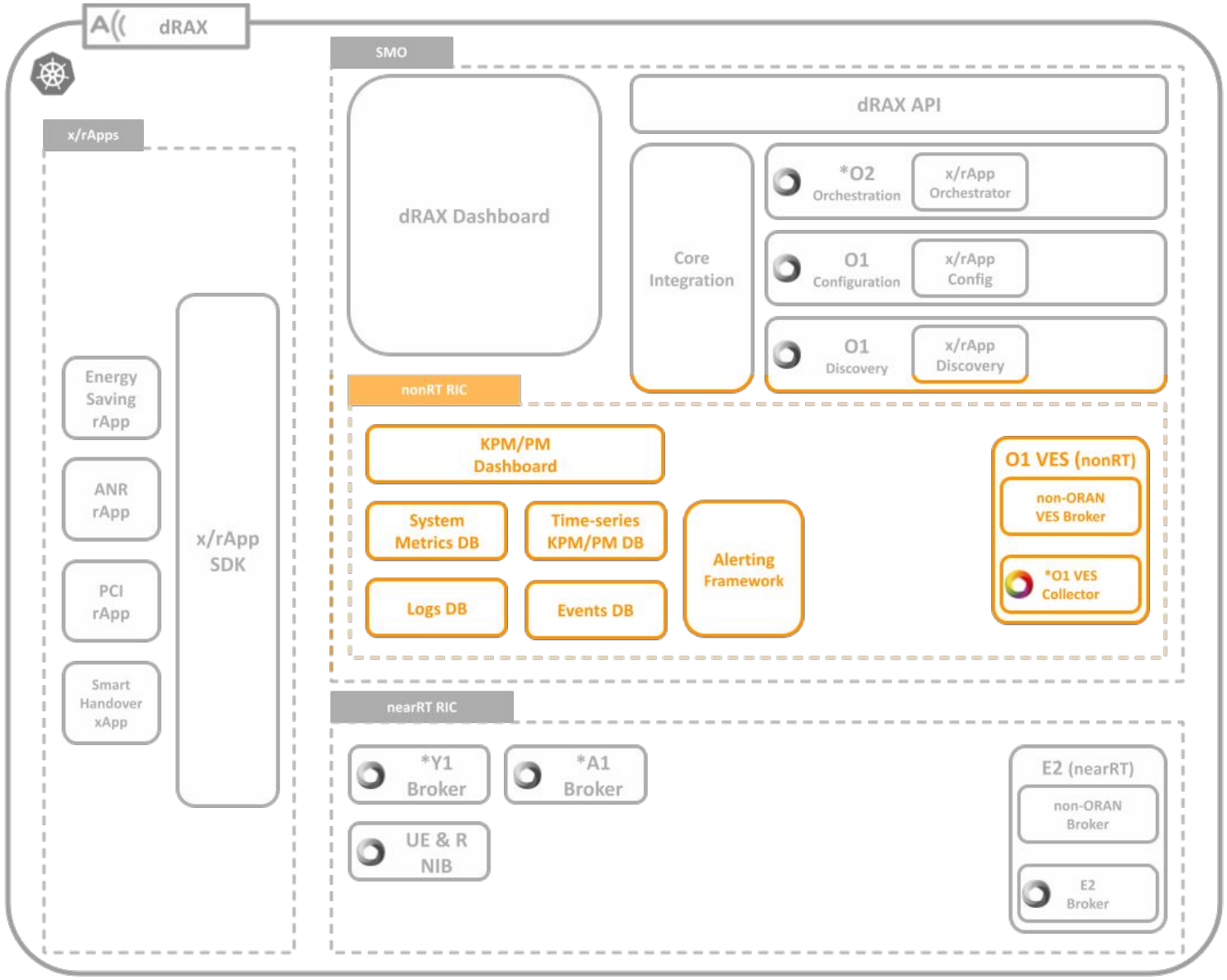
O1 Orchestration:
Onboarding & lifecycle management of x/rApps

O1 Configuration:
Configuration of x/rApps

O1 Discovery:
Discovery of x/rApps



dRAX nonRT RIC



O1 3GPP PM Streaming Collector:
Ability to receive nonRT data from RAN from ORAN and Proprietary Interfaces

dRAX Databases:
Systems metrics, KPMs, Events and Logs databases for historical purposes

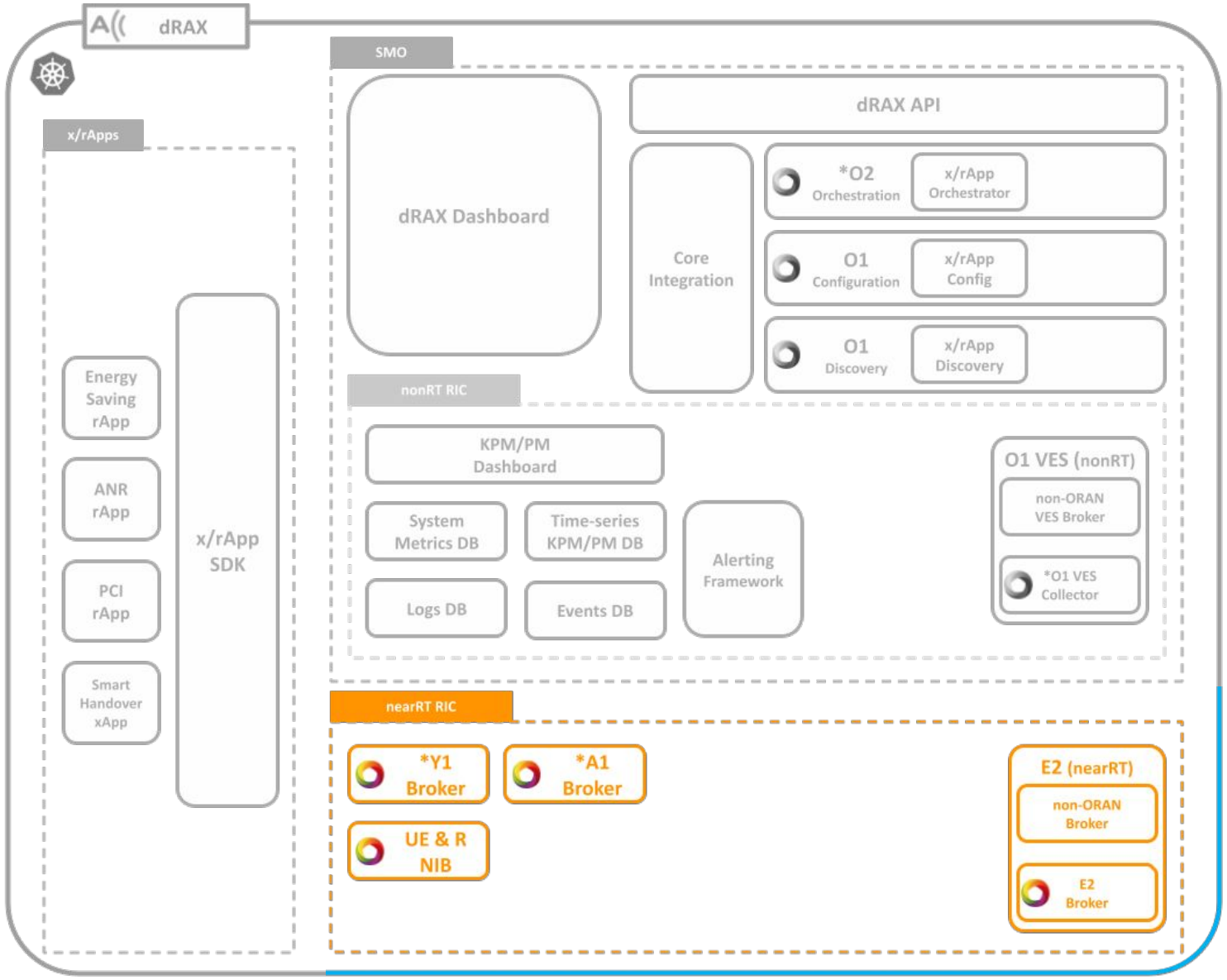
dRAX KPM Dashboard:
Grafana that visualises data from the dRAX Databases

dRAX Alerting Framework:
Create generic Alarms based on any of the dRAX Databases

A1 and O1:
Internal communication to Accelleran nearRT RIC for A1 and O1 messages

Features:
O1 VES KPM/PM Monitoring
Events Monitoring
System health / Performance Monitoring
Logs Monitoring
Alerting Framework

dRAX nearRT RIC



E2 (nearRT) Broker:
Support for the official E2 Protocol via the E2 Broker

ORAN SDK Approach:
E2 Broker handles the SCTP and ASN.1 encoding

E2 Versions:

- E2 AP: v5.00
- E2 SM: v5.00
- E2 SM KPM: v5.00
- *E2 SM RC: v6.00
- E2 SM CCC: v4.00 (JSON)

Y1 Interface:

- Y1-like functionality is supported via proprietary interface
- Enabling extraction of Data from the RIC to outside world
- Extending functionality to also allow Data Enrichment from outside world to RIC

A1 Interface:
A1-like functionality is supported via proprietary interface (Policy Management)

O1 Interface:
O1-like Data Enrichment capabilities supported via proprietary interface

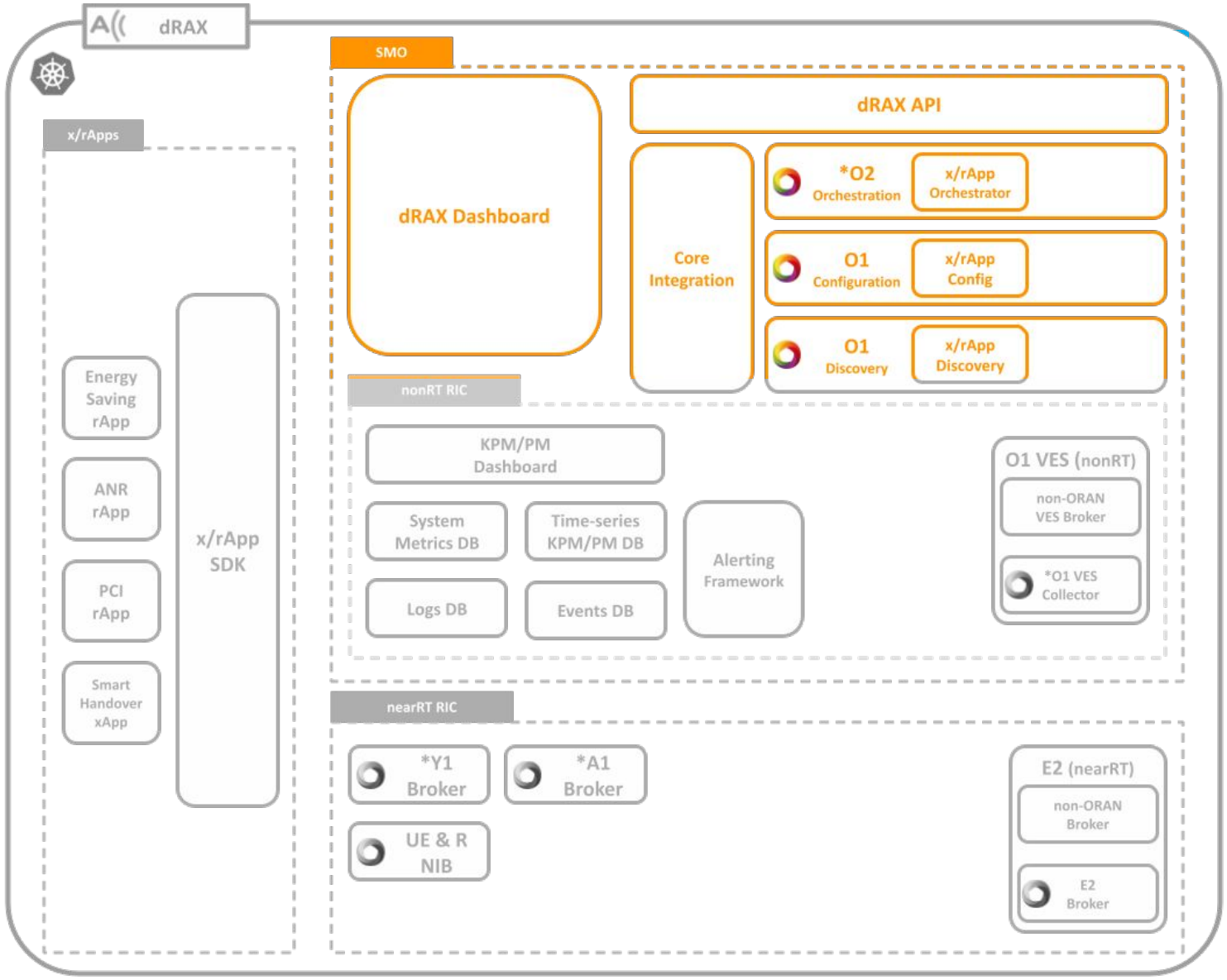
UE NIB & R NIB:

- Only Accelleran RAN supported at the moment (update in development to support E2 for any RAN)
- CU & DU information stored
- UE information stored
- With Core Integration - SUPI/IMSI for UEs available as well

You don't have to wait:
The RIC solution is built on open standards but also enables functionality ahead of formal standards.

*A selected set of E2 RC SM messages are supported, with ongoing enhancements planned.

dRAX Core Integration



Core integration in the RIC

Currently supported Cores: Open5GS

Expose the SUPI/IMSI of connected UEs

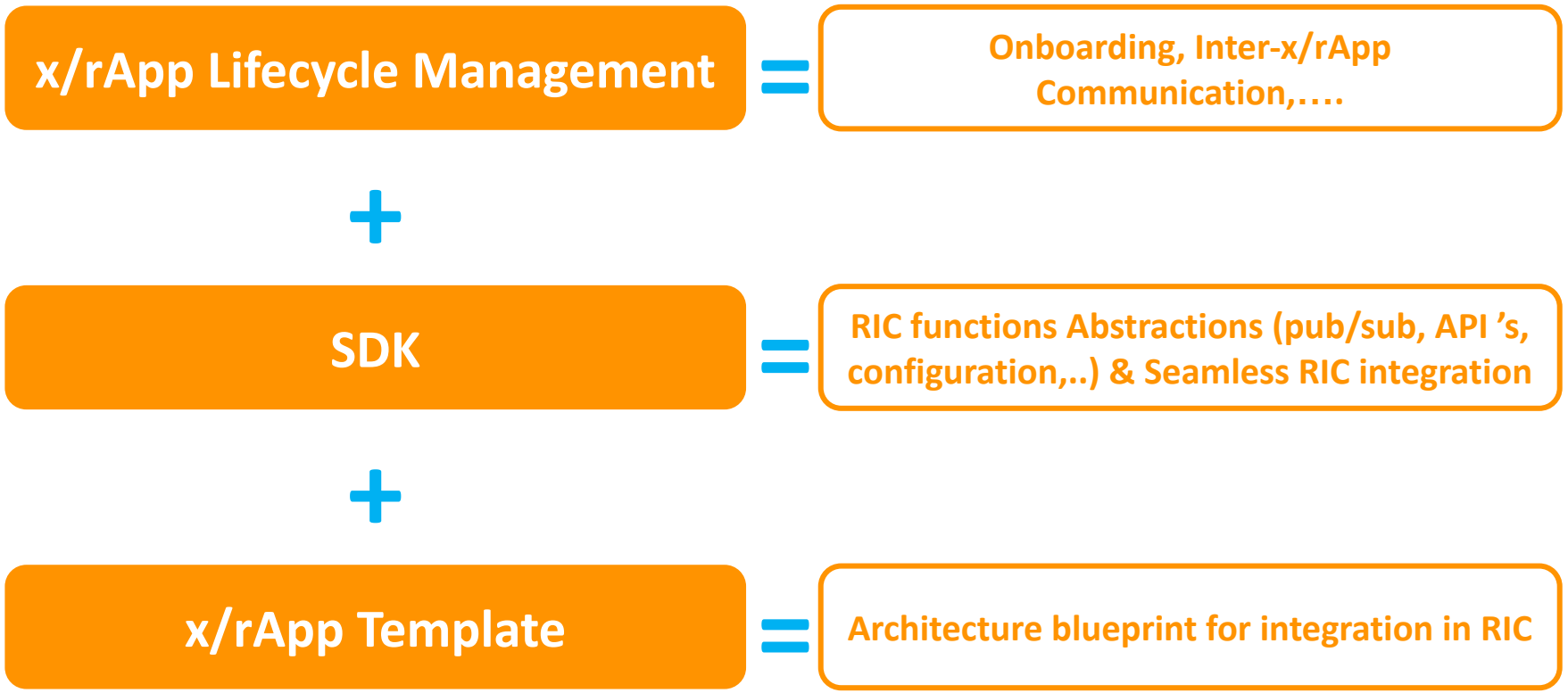
Enabled unique identification & tracking of UEs

Expose UE information to RIC State Database

x/rApps can consult UE info, including SUPI/IMSI

x/rApp Development Framework

dRAX x/rApp Developer Framework



Link to dRAX x/rApp Dev Guide

x/rApp Features

x/rApp Dev Env

- Sandbox environment for Developing x/rApps
- All pre-requirements pre-installed
- Ready to code workflow

Lifecycle Management

- Lifecycle management provided by platform itself
- Productizing x/rApps
- Onboarding can be done via API or dRAX Dashboard

xApp SDK

- Abstracting the RIC Interfaces
- Seamless integration with RIC
- Easy development

E2

- E2 support via the E2 Broker
- E2 SDK Abstractions to interfaces with E2 Broker
- No need to handle SCTP and ASN.1 encoding in x/rApp
- Ability to use other E2 SMs by receive the E2 AP decoded message from the E2 Broker

xApp Configuration

- Productized x/rApp Configuration
- x/rApp Configuration exposed programmatically using APIs
- x/rApp Configuration integrated visually in the dRAX Dashboard

Alarm Management

- Integrated Alarmin Framework
- Ability to monitor and trigger Alarms

Inter-x/rApp Communication

- Simplified inter-x/rApp communication provided by platform

Events Monitoring

- Integrated Events handling
- Ability to visualize Events on dRAX Dashboard

Logs Monitoring

- Native support for monitoring Logs of x/rApps
- Integrated Logs monitoring on Dashboards

Documentation

- Fully documented x/rApp Dev Guide and SDK
- x/rApp templates as Hello World examples

xApp APIs

- Modular and extendable API creation in the x/rApps
- Ability to support custom APIs

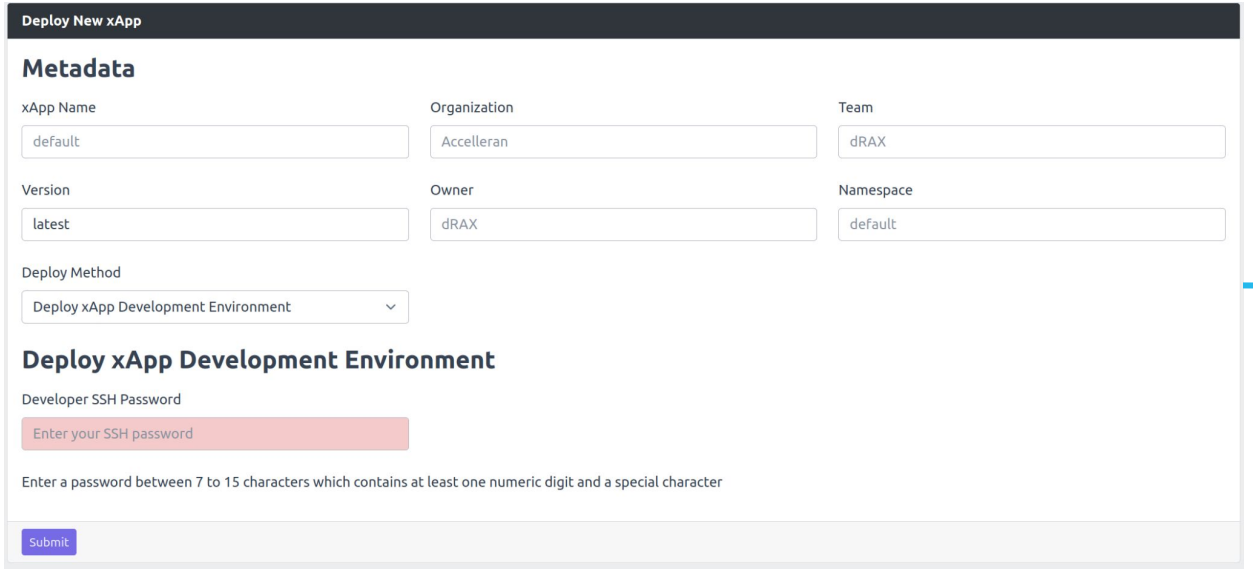
Performance Monitoring

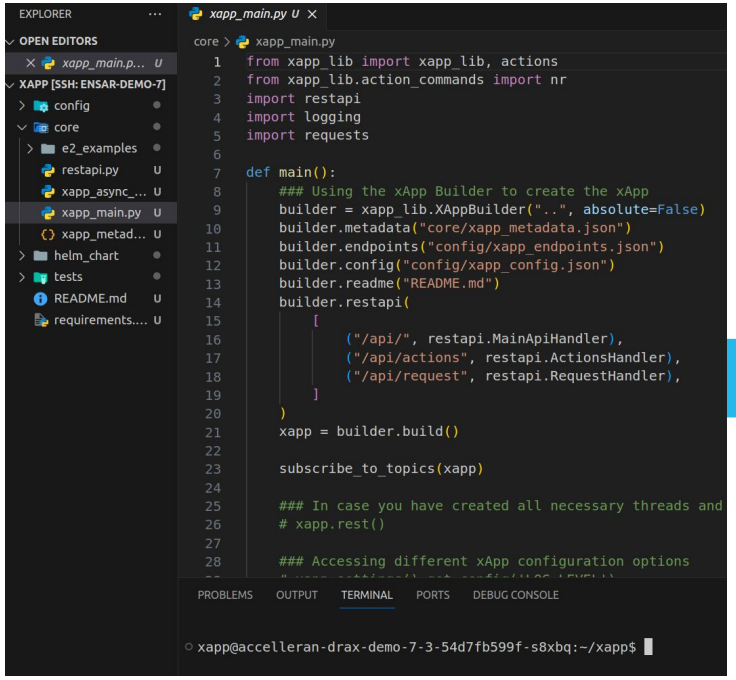
- Performance monitoring of x/rApp provided by platform

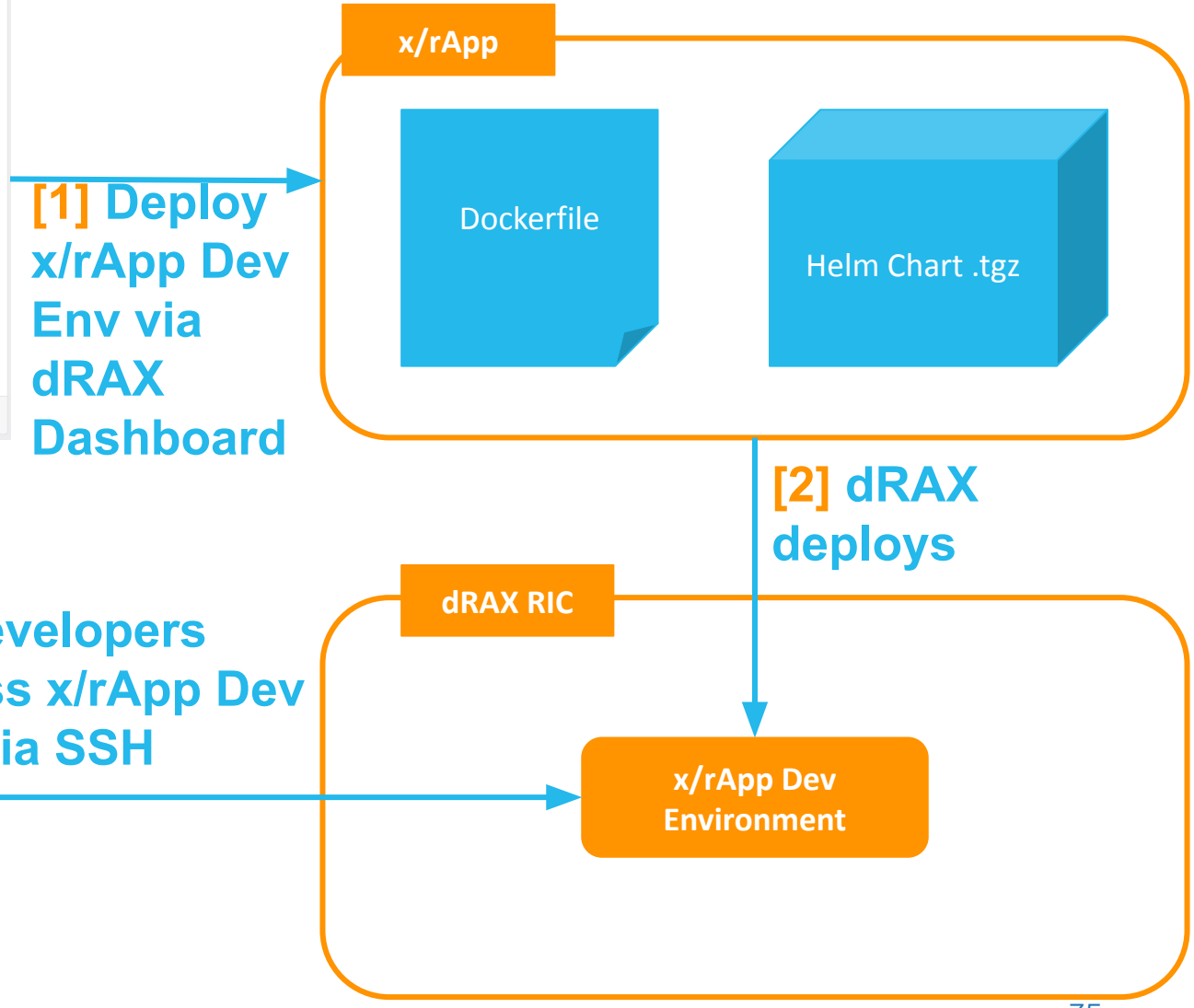
x/rApp Dashboards

- Ability to create custom x/rApp Dashboards
- Visualise Logs, Performance monitoring, Events, KPMs, and other data from x/rApps

x/rApp Development Environment







RIC Data & Controls

Accelleran CU - KPMs & PMs

Link to Accelleran CU KPMs & PMs documentation

- DU beacon info - This message gives information about the DUs connected to the CU-CP and their operational state (including [3GPP 38.473] served cell information).
- CU-CP beacon info - This message gives information about the operation state of the CU-CP.
- CU-UP beacon info - This message gives information about the CU-UPs connected to the CU-CP and their operation state.
- AMF beacon info - This message gives information about the AMFs connected to the CU-CP.
- UE Measurements - These measurements contain the RSRP, RSRQ & SINR periodic measurements from serving cell.

- CU-UP PM Counters:
  - DL_GTP_THP
  - DL_GTP_THP_UE_QOS_SNSSAI_PLMN
  - UL_GTP_THP
  - UL_GTP_THP_UE_QOS_SNSSAI_PLMN
  - DRB_PACKET_LOSS_UL_QOS
  - DRB_PACKET_LOSS_UL_SNSSAI
  - DRB_PACKET_LOSS_UL_QOS_SNSSAI_PLMN
  - DRB_PDCP_PACKET_DROP_RATE_DL_QOS
  - DRB_PDCP_PACKET_DROP_RATE_DL_SNSSAI
  - DRB_PDCP_PACKET_DROP_RATE_DL_QOS_SNSSAI_PLMN
  - AVERAGE_DELAY_DL_IN_CUUP_QOS
  - AVERAGE_DELAY_DL_IN_CUUP_SNSSAI
  - AVERAGE_DELAY_DL_IN_CUUP_QOS_SNSSAI_PLMN
  - DISTRIBUTION_OF_DELAY_DL_IN_CUUP_QOS
  - DISTRIBUTION_OF_DELAY_DL_IN_CUUP_SNSSAI
  - DISTRIBUTION_OF_DELAY_DL_IN_CUUP_QOS_SNSSAI_PLMN

- CU-CP PM Counters:
  - RRC_CONNECTION_EST_ATTEMPT
  - RRC_CONNECTION_SUCC
  - REQ_INTRA_GNB_HANDOVER_EXECUTIONS
  - SUCC_INTRA_GNB_HANDOVER_EXECUTIONS
  - MEAN_NUMBER_OF_RRC_CONNECTIONS
  - MAXIMUM_NUMBER_OF_RRC_CONNECTIONS
  - NUMBER_OF_PDU_SESSIONS_REQUESTED_TO_SETUP
  - NUMBER_OF_PDU_SESSIONS_FAILED_TO_SETUP
  - SS_RSRP_DISTRIBUTION_PER_SSB
  - REQ_INTER_GNB_HANDOVER_PREPARATIONS
  - SUCC_INTER_GNB_HANDOVER_PREPARATIONS
  - FAILED_INTER_GNB_HANDOVER_PREPARATIONS
  - REQ_INTER_GNB_HANDOVER_ALLOCATIONS
  - SUCC_INTER_GNB_HANDOVER_ALLOCATIONS
  - FAILED_INTER_GNB_HANDOVER_ALLOCATIONS
  - REQ_INTER_GNB_HANDOVER_EXECUTIONS

How to get CU KPMs & PMs

```python
def subscribe_to_topics(xapp):
    ### Subscribing to topics on the dRAX RIC Databus (Kafka)
    xapp.kafka().subscribe([
        'accelleran.drax.4g.ric.raw.messages',
        'accelleran.drax.4g.ric.raw.ue_measurements',
        'accelleran.drax.4g.ric.raw.service_discovery',
        'accelleran.drax.4g.ric.raw.network_state',
        'accelleran.drax.5g.ric.raw.ue_measurements',
        'accelleran.drax.5g.ric.raw.cu_state',
        'accelleran.drax.5g.ric.processed.pm_counters',
        'accelleran.drax.5g.ric.raw.pm_counters',
        'accelleran.drax.5g.ric.raw.messages',
        'accelleran.drax.5g.ric.raw.ran_control_response',
        # 'accelleran.drax.5g.ric.o1.ves',
        # 'accelleran.drax.all.ric.processed.anr',
        ])
```

```python
def process_data(xapp):
        # How to read JSON data from the dRAX RIC Databus (Kafka)
        (topic, data) = xapp.kafka().json().recv_message()

        ### Example how to read generic data from dRAX NATS Databus
        # (topic, data) = xapp.nats().recv_data()

        ### Example 1: Just logging all the messages received from the dRAX Databus
        log_message("Received message on the dRAX RIC Databus topic [{topic}]: {data}".format(
            topic=topic,
            data=data
        ))
```

Accelleran CU Events

- F1-SEIZE-REQUEST
  - Topic: F1-SEIZE-REQUEST
  - Description: Sent when a DU connects to the CU over F1-C
- F1-SCTP-ASSOC-FAILURE-IND.DU
  - Topic: F1-SCTP-ASSOC-FAILURE-IND.DU=*
  - Description: Sent when a DU disconnects from the CU on F1-C
- F1-ASSOC-CLOSE-REQ.DU
  - Topic: F1-ASSOC-CLOSE-REQ.DU=*
  - Description: Sent when the CU closes the F1-C connection to a DU
- UE-RESET-INDICATION.UE
  - Topic: UE-RESET-INDICATION.UE=*
  - Description: Sent when the CU resets a UE for various reasons
- UE-RELEASE-INDICATION
  - Topic: UE-RELEASE-INDICATION
  - Description: Sent when a UE is released for whatever reason
- UE-INFO-UPDATE-INDICATION
  - Topic: UE-INFO-UPDATE-INDICATION
  - Description: Sent when a UE connects or any of its attributes change
- DSC-PMCTRL-DRB-SETUP-SUCCESS
  - Topic: DSC-PMCTRL-DRB-SETUP-SUCCESS
  - Description: A DRB was successfully set up
- DSC-PMCTRL-DRB-RELEASE
  - Topic: DSC-PMCTRL-DRB-RELEASE
  - Description: A DRB was released
- DSC-PMCTRL-NG-HANDOVER-EXECUTION-SUCCESS
  - Topic: DSC-PMCTRL-NG-HANDOVER-EXECUTION-SUCCESS
  - Description: An NGAP HO was successfully executed
- DSC-PMCTRL-NG-HANDOVER-EXECUTION-FAILURE
  - Topic: DSC-PMCTRL-NG-HANDOVER-EXECUTION-FAILURE
  - Description: An NGAP HO has failed
- F1-SCTP-ASSOC-FAILURE-IND.DU
  - Topic: F1-SCTP-ASSOC-FAILURE-IND.DU=*
  - Description: Sent when a DU disconnects from the CU on F1-C

How to monitor CU Events

```
### Example: Monitoring dRAX Events
# List of dRAX Events:
drax_events = [
    'F1-SEIZE-REQUEST',
    'F1-SCTP-ASSOC-FAILURE-IND.DU',
    'F1-ASSOC-CLOSE-REQ.DU',
    'UE-RESET-INDICATION.UE',
    'UE-RELEASE-INDICATION',
    'UE-INFO-UPDATE-INDICATION',
    'DSC-PMCTRL-DRB-SETUP-SUCCESS',
    'DSC-PMCTRL-DRB-RELEASE',
    'DSC-PMCTRL-NG-HANDOVER-EXECUTION-SUCCESS',
    'DSC-PMCTRL-NG-HANDOVER-EXECUTION-FAILURE'
]
# Filter dRAX Events and log them
for drax_event in drax_events:
    if drax_event in data['topic']:
        log_message("Received [{drax_event}] dRAX Event: {data}".format(
                drax_event=drax_event,
                data=data
        ))
```

Accelleran CU - RAN Control

- Handover - Link to Handover control documentation

- Admission control - Link to Admission Control documentation
  - UE Admission Control
  - PDU Session Admission Control

Accelleran DU - KPMs & PMs

Accelleran DU - KPMs & PMs Documentation

- E2 KPMs:
  - CQI
  - RSRP
  - RSRP
  - DRB.UEThpDl - DL throughput
  - DRB.UEThpUl - UL throughput
  - DRB.RlcPacketDropRateDl - UL packet success rate
  - DRB.PacketSuccessRateUlgNBUu - RLC DL packet drop rate
  - DRB.RlcSduTransmittedVolumeDL - RLC DL transmitted SDU volume
  - DRB.RlcSduTransmittedVolumeUL - RLC UL transmitted SDU volume
- O1 PMs:
  - - cqi
  - - ri
  - - dl_mcs
  - - dl_prbs_used
  - - dl_brate
  - - dl_nof_ok
  - - dl_nof_nok
  - - dl_bs
  - - pusch_snr_db
  - - pucch_snr_db
  - - ul_mcs
  - - ul_prbs_used
  - - ul_brate
  - - ul_nof_ok
  - - ul_nof_nok
  - - bsr
  - - last_ta
  - - last_phr
  - - tx_num_sdus
  - - tx_num_sdu_bytes
  - - tx_num_dropped_sdus
  - - tx_num_discarded_sdus
  - - tx_num_discard_failures
  - - tx_num_pdus
  - - tx_num_pdu_bytes
  - - rx_num_sdus
  - - rx_num_sdu_bytes
  - - rx_num_pdus
  - - rx_num_pdu_bytes
  - - rx_num_lost_pdus
  - - rx_num_malformed_pdus

How to get DU KPMs & PMs

```python
def subscribe_to_topics(xapp):
    ### Subscribing to topics on the dRAX RIC Databus (Kafka)
    xapp.kafka().subscribe([
        'accelleran.drax.4g.ric.raw.messages',
        'accelleran.drax.4g.ric.raw.ue_measurements',
        'accelleran.drax.4g.ric.raw.service_discovery',
        'accelleran.drax.4g.ric.raw.network_state',
        'accelleran.drax.5g.ric.raw.ue_measurements',
        'accelleran.drax.5g.ric.raw.cu_state',
        'accelleran.drax.5g.ric.processed.pm_counters',
        'accelleran.drax.5g.ric.raw.pm_counters',
        'accelleran.drax.5g.ric.raw.messages',
        'accelleran.drax.5g.ric.raw.ran_control_response',
        # 'accelleran.drax.5g.ric.o1.ves',
        # 'accelleran.drax.all.ric.processed.anr',
        ])
```

```python
def process_data(xapp):
        # How to read JSON data from the dRAX RIC Databus (Kafka)
        (topic, data) = xapp.kafka().json().recv_message()

        ### Example how to read generic data from dRAX NATS Databus
        # (topic, data) = xapp.nats().recv_data()

        ### Example 1: Just logging all the messages received from the dRAX Databus
        log_message("Received message on the dRAX RIC Databus topic [{topic}]: {data}".format(
            topic=topic,
            data=data
        ))
```

Accelleran CU-CP - RAN

[7]
```
"connected-gnb-du": [
  {
    "@xmlns": "http://accelleran.com/ns/yang/accelleran-connected-gnb-du",
    "gnb-du-id": "2",
    "gnb-du-name": null,
    "served-cell": {
      "plmn-identity": "99999",
      "nr-cell-identity": "18",
      "nr-pci": "502",
      "ranac": "0",
      "tac-5gs": "65536",
      "served-plmn-slices": {
        "plmn-id": "99999",
        "s-nssai": {
          "sst": "embb",
          "sd": "0"
        }
      }
    }
  },
```

[1]
```
{
        "plmn-id": "99999",
        "gnb-id": "1",
        "gnb-id-length": "32",
        "gnb-cu-cp-name": "accelleran-gnb",
        "admin-state": "unlocked",
        "allowed-integrity-algorithm": [...],
        "allowed-ciphering-algorithm": "nea0",
        "handover-control": {...},
        "rrc-inactive-state": {"t380": "min5"...},
        "ng-cell": [...],
        "operator": {"operator-id": "operator-1"...},
        "connected-gnb-du": [...]
}
```

[5]
```
"ng-cell": [
  {
    "plmn-identity": "99999",
    "nr-cell-identity": "17",
    "local-cell-lists": {
      "local-nr-cell": {
        "plmn-identity": "99999",
        "nr-cell-identity": "18",
        "physical-cell-id": "502",
        "blacklisted": "false",
        "dl-CarrierFreq": "649440",
        "duplex-mode": "tdd",
        "ssb-subcarrier-spacing": "kHz30",
        "smtc1": {
          "smtc-periodicity": "sf10",
          "smtc-offset": "0",
          "duration": "2"
        }
      }
    }
  },
```

[4]
```
"rrc-inactive-state": {
  "t380": "min5",
  "max-rna-fail": "3",
  "ran-paging-cycle": "rf128",
  "paging-repeat-timer": "5",
  "max-paging-repeats": "3",
  "inactivity-timer": "10"
},
```

[6]
```
"operator": {
  "operator-id": "operator-1",
  "ng-c-link": {
    "dest-address": "10.55.5.4",
    "default-amf": "true",
    "op-state": "enabled"
  }
},
```

[3]
```
"handover-control": {
  "report-config-a3-rsrp": {
    "a3-offset": "2",
    "trigger-quantity": "rsrp",
    "hysteresis": "2",
    "time-to-trigger": "320"
  }
},
```

[2]
```
"allowed-integrity-algorithm": [
  "128-nia2",
  "128-nia1",
  "nia0"
],
```

Accelleran CU-UP - RAN Configuration

[1]

```
{
      "gnb-cu-up": {
        "@xmlns": "http://accelleran.com/ns/yang/accelleran-gnb-cu-up",
        "gnb-cu-up-name": "accelleran-cu-up-1",
        "gnb-cu-up-id": "1",
        "admin-state": "unlocked",
        "e1-link": {"dest-address": "10.55.5.30"...},
        "supported-plmn-slices": [...]
      }
}
```

[2]

```
"e1-link": {
  "dest-address": "10.55.5.30"
},
```

[3]

```
"supported-plmn-slices": [
  {
    "plmn-id": "99999",
    "s-nssai": [
      {
        "sst": "embb",
        "sd": "16777215"
      },
      {
        "sst": "urllc",
        "sd": "16777215"
      }
    ]
  },
```

Accelleran DU - RAN Configuration

[1]


```
{
  "gnb_du_id": "1",
  "admin_state": "unlocked",
  "cucp_instance_id": "cucp-1",
  "local_cell_id": "1",
  "tac": "1",
  "pci": "501",
  "ru_name": "ru-1",
  "frequency_band": "78",
  "arfcn_dl": "650112",
  "bandwidth_dl": "40",
  "bandwidth_ul": "40",
  "maximum_eirp_dbm": "35",
  "plmn_slice_list": [
    {
      "plmn_id": "99999",
      "snssai_list": [
        {
          "sst": "embb",
          "sd": "16777215"
        }
      ]
    }
  ]
}
```

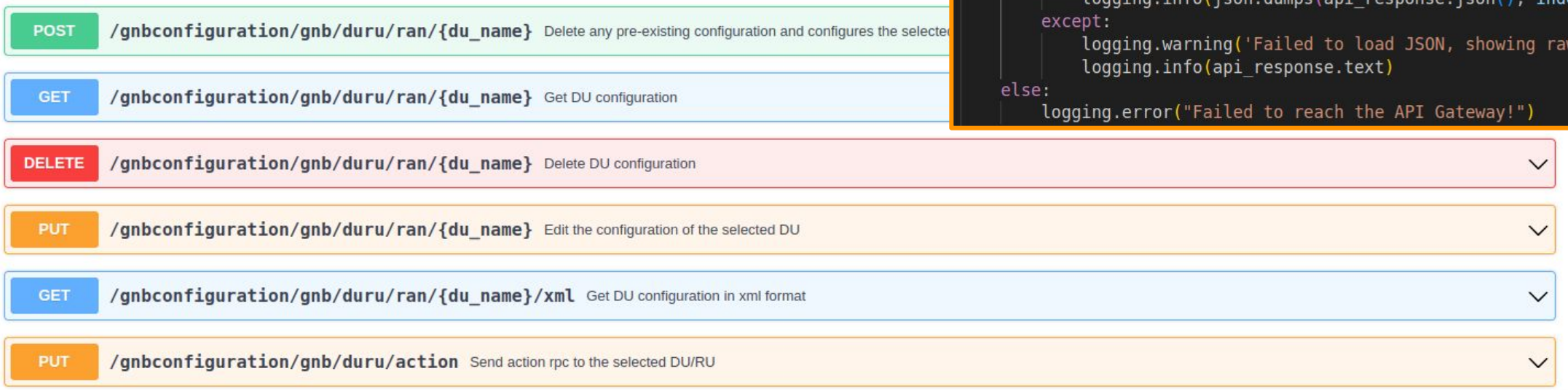
How to do O1 Configuration of CU & DU

[dRAX API - Documentation Link]

```python
def main():
    endpoint = '/gnbconfiguration/gnb/duru/ran/du-1'
    api_url = actions.create_api_url(xapp, endpoint)
    api_response = requests.get(api_url)
    if api_response.status_code == 200:
        try:
            logging.info(json.dumps(api_response.json(), indent=2))
            config = api_response.json()
        except:
            logging.warning('Failed to load JSON, showing raw content of API response:')
            logging.info(api_response.text)
    else:
        logging.error("Failed to reach the API Gateway!")

    config['admin_state'] = "unlocked"

    endpoint = '/gnbconfiguration/gnb/duru/ran/du-1'
    api_url = actions.create_api_url(xapp, endpoint)
    api_response = requests.post(api_url, json.dumps(config))
    if api_response.status_code == 200:
        try:
            logging.info(json.dumps(api_response.json(), indent=2))
        except:
            logging.warning('Failed to load JSON, showing raw content of API response:')
            logging.info(api_response.text)
    else:
        logging.error("Failed to reach the API Gateway!")
```



Demo

Q & A

www.6g-leader.eu

Linkedin.com/6g-leader

Youtube.com/6g-leader

6G LEADER

Thank you for your attention

Accelleran

info@accelleran.com
german.castellanos@accelleran.com
ensar.Zeljkovic@accelleran.com

Funded by the European Union

6GSNS

Funded by the European Union. The project is supported by Smart Networks and Services Joint Undertaking (SNS JU) and its members. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or SNS JU. Neither the European Union nor the granting authority can be held responsible for them.