MobSys Lab 2: **Open RAN Basics**

Navid Nikaein, Alireza Mohammadi, Chieh-Chun Chen, and Zaineb Benamar

2023-2024

Important Note

While copying text from the PDF, please make sure you respect the spacings. The spaces are not copyable from the PDF, so you should type them manually. For ease of use, a copy of all the code snippets is provided separately in the `code.sh` file.

After one week, a sample file for Wireshark PCAPs would be uploaded to Moodle. Students who cannot finish the lab on time, could use this PCAP to answer the questions at the end of the lab to at least get the grades for the questions.

1 Prerequisites

First, login again to your user account by running the following command:

```
$ gcloud auth login
```

This command will open a browser window and ask you to login to your Google account. If the browser did not open automatically, you can copy the URL from the terminal and paste it in your browser. After logging in, you will be asked to grant access to the Google Cloud SDK.

Now you need to retrieve the credentials for the GKE cluster of your group. To do so, run the following command:

```
$ gcloud container clusters get-credentials ors-cluster{group-id} \
    --region europe-west6 --project comsyslab
```

You should replace the `{group-id}` with your group number.

Now you should be able to use the CLI to interact with the platform. To test it out run the following command:

```
$ cli extract infra
```

In the result you should see a brief description of the infrastructure of the whole cluster, including the different nodes, their properties, and the associated radio components. This time you should see only two nodes.

2 Background

For doing this lab you need to be familiar with the following three concepts:

1. F1 Protocol Stack
2. Open RAN Architecture and Interfaces
3. xApp Principles

2.1 F1 Protocol Stack

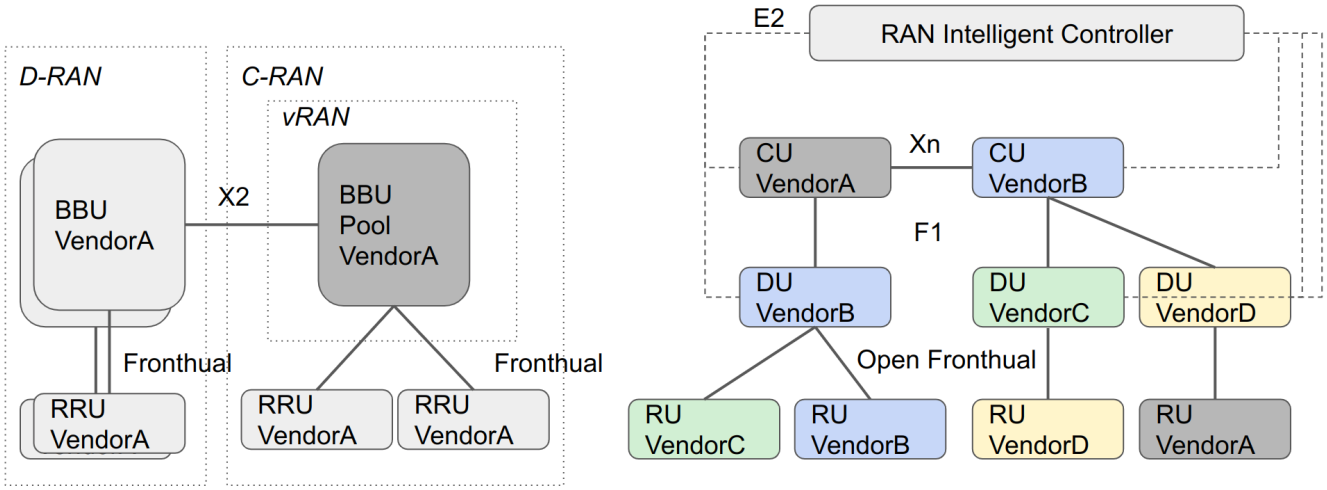

Figure 1: Traditional RAN v.s Open RAN

Traditionally, especially in 3G and 4G (LTE), a Radio Access Network (RAN) is composed of three primary components:

1. **Baseband Unit (BBU)**: Responsible for managing the entire base station, the BBU handles tasks such as operating and maintenance procedures, signaling processing, security implementation, resource optimization, and error detection. It plays a crucial role in determining the overall system capacity and focuses only on the computation done on the signal in a baseband frequency domain.
2. **Remote Radio Unit (RRU)**: The RRU interfaces with an antenna on one end and the BBU on the other. Through the CPRI (Common Public Radio Interface) interface, it connects to the BBU and performs the conversion of RF signals into data signals and vice versa. Additionally, the RRU carries out filtering and amplification of RF signals, thereby influencing the system's coverage.
3. **Antenna**: Functioning to convert electrical signals into electromagnetic waves, the antenna establishes a wireless interface with cell phones, facilitating the transmission and reception of RF signals. The antenna's characteristics play a key role in determining the shape of the coverage area.

Based on these components, the following RAN architectures have emerged in the following order:

D-RAN (Distributed or traditional RAN) : The D-RAN was based on the co-location of RRU and BBU at each cell site. All radio functions at every cell site are distributed, and connectivity to the core network is established through backhaul links, which are typically fiber optic cables. The D-RAN architecture is the term used to describe the traditional RAN architecture.

C-RAN (Centralized or cloud RAN) : C-RAN structure groups the BBUs from several sites into a pool located in one location, and the cell site only has the antenna and the RRU. This centralization of the BBU pool results in a new interface called fronthaul that enables multiplexing gains via interference cancellation and coordinated multipoint (CoMP) transmission. The fact that the BBUs use a hardware pool motivates the use of the term *cloud* for this RAN architecture. Do not confuse this with the cloud computing concept.

3GPP later improved this concept by defining the F1 split of the BBUs into the CU (Central Unit) and DU (Distributed Unit), where the protocol stack is split into two parts, as shown in Figure 2. This split further enables the CoMP transmission and interference cancellation gains. Naturally, the CU is placed closer to the core network and the DUs are located closer to the cell sites, hence defining the fronthaul and midhaul interfaces. It should be noted that the gains of the F1 split are only viable if multiple DUs are connected to a single CU and the midhaul interface provides sufficient bandwidth.

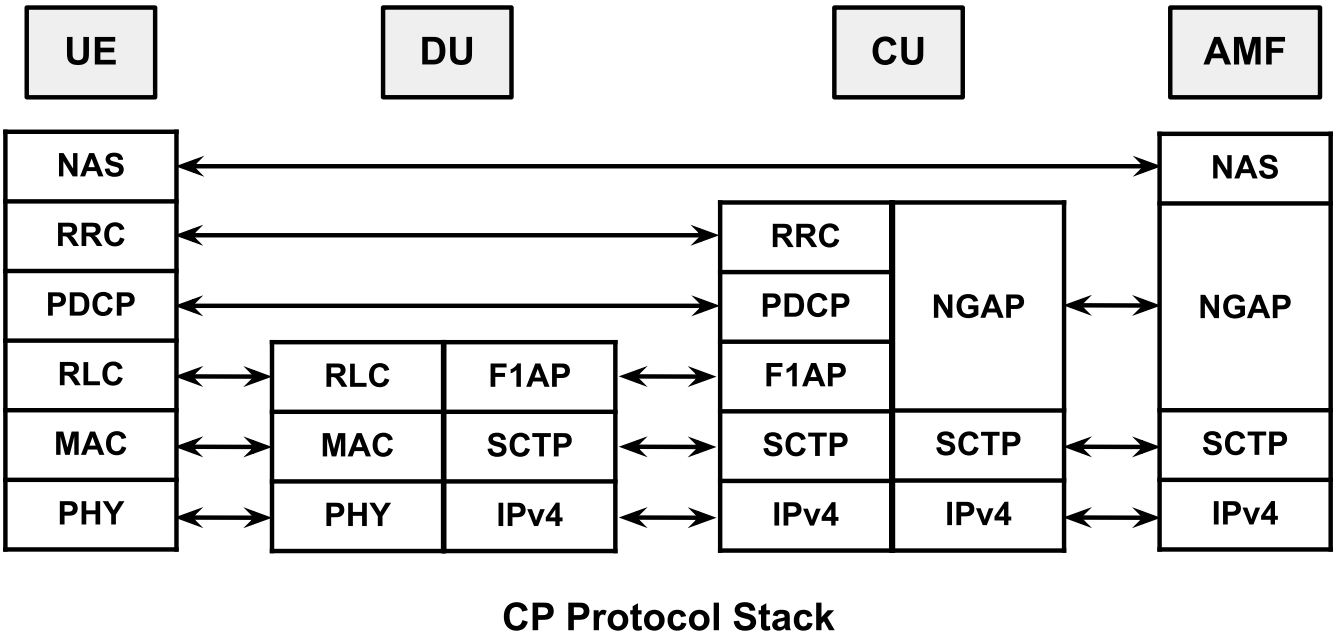

Figure 2: CP Protocol Stack for F1-splitted gNBs.

V-RAN (virtual RAN) : V-RAN decouples the software from hardware by virtualizing Network Functions (NFs). The difference between V-RAN and C-RAN is that C-RAN uses the vendor's proprietary hardware and software while V-RAN uses virtualized proprietary software running on top of Commercial Off-The-Shelf (COTS) servers. This is how we are essentially able to run the lab software on the Google Cloud Platform (GCP) servers.

Open RAN : While V-RAN frees the operators from reliance on specific hardware vendors, the need for virtualization technologies for CU and DU from the same vendor persists due to proprietary software. Open RAN addresses this by introducing open interfaces in fronthaul, mid-haul and backhaul, allowing customers to mix components from different vendors for Open RAN networks.

Based on the concept of Open RAN, the O-RAN Alliance was formed in 2018 by a group of operators to drive the adoption of open interfaces and virtualization of RAN components. The O-RAN architecture shown in Figure 3 is designed based on three principles:

1. Open interfaces for multi-vendor interoperability
2. Intelligent RAN control
3. Virtualization of NFs

2.2 Open RAN Architecture and Interfaces

We further discuss the Open RAN architecture and interfaces in this section.

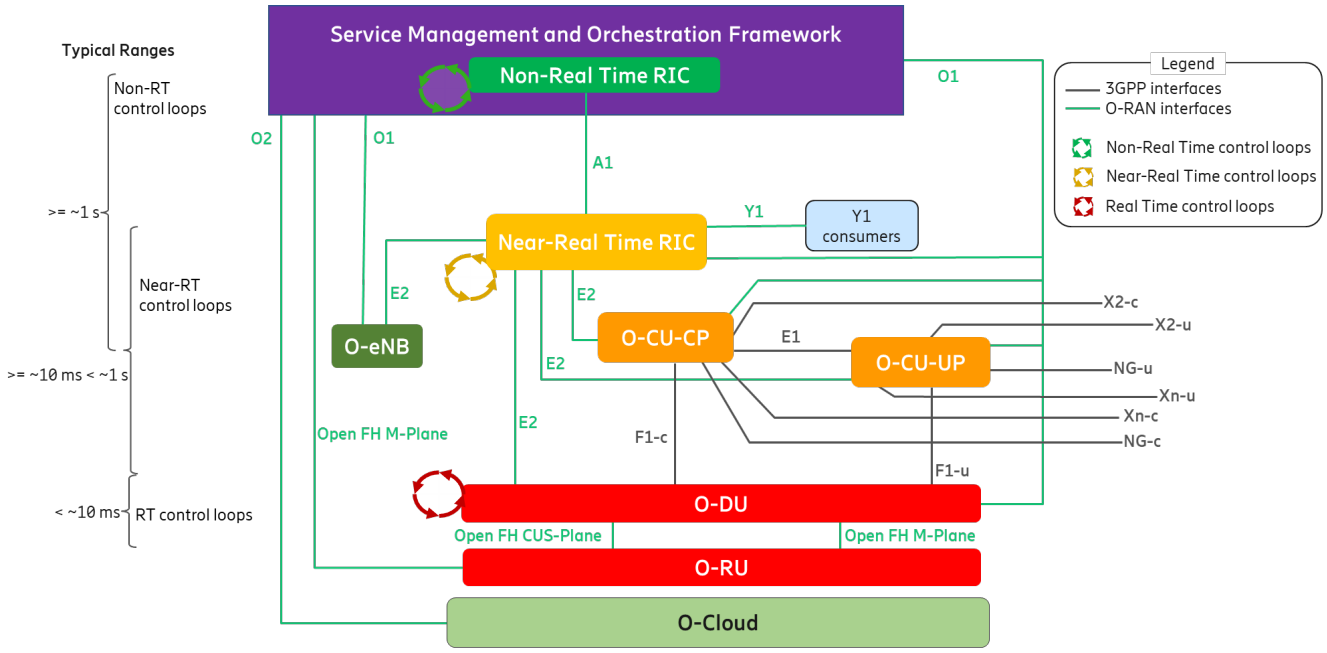

Figure 3: O-RAN Architecture. (Ref: O-RAN Architecture Description v10.0)

In Figure 3, there are two type of interface: (1) 3GPP interfaces (e.g., F1, E1, X2 and NG) and (2) O-RAN interfaces:

- **E2** is an interface between Near-Real-Time RIC (NearRT-RIC) and E2-Nodes (e.g., DU, CU-UP, CU-CP and eNB), and it allows NearRT-RIC to control and monitor the selected RAN functions in the E2-Nodes, facilitating dynamic optimization and management. (Ref: O-RAN E2 General Aspects and Principles)

- **Open Fronthaul** is an interface between DU and RU with a lower layer functional split 7.2x (Ref: O-RAN Fronthaul Control, User and Synchronization Plane Specification v07.02)
- **O1** is an interface between SMO and network elements (e.g., NearRT-RIC, DU, CU-UP, etc.), responsible for deploying and configuring the network elements. (Ref: O-RAN Operations and Maintenance Interface Specification 11.0)
- **A1** is an interface between NearRT-RIC and Non-Real-Time RIC (NonRT-RIC), enabling the exchange of policy management and enrichment information between NearRT-RIC and NonRT-RIC. (Ref: O-RAN A1 interface: General Aspects and Principles)
- the other interfaces, such as O2 and Y1 are out of scope for the current lab.

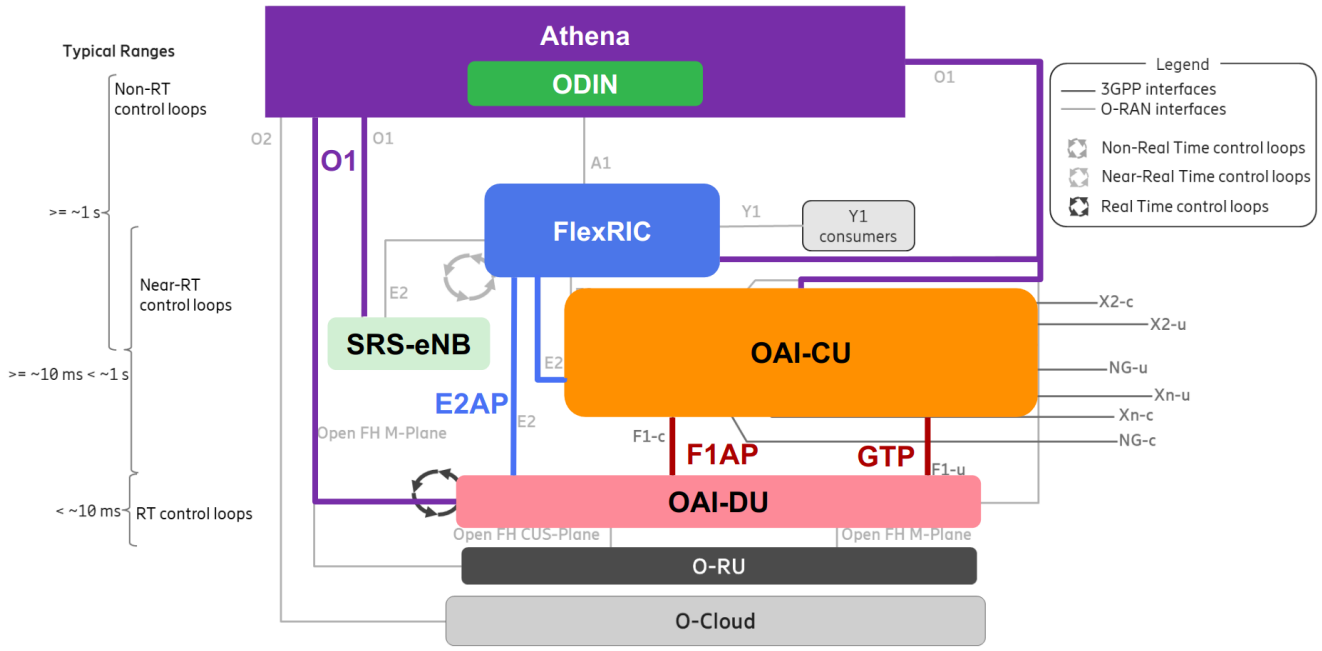

Figure 4: Open RAN Lab Architecture.

For our lab, the network deployment is as shown in Figure 4, which uses FlexRIC[1] as NearRT-RIC, ODIN as NonRT-RIC and Athena as Service Management and Orchestration (SMO). E2-Nodes, including CU and DU provided by OpenAirInterface, will setup E2 connections (SCTP mechanism) with NearRT-RIC.

2.3 xApp Principles

- xApp is an application running on top of NearRT-RIC to control and monitor specific RAN functions in connected E2-Nodes via the RIC report and control service. It is independent of NearRT-RIC and can be provided by any third party.

[1] https://gitlab.eurecom.fr/mosaic5g/flexric

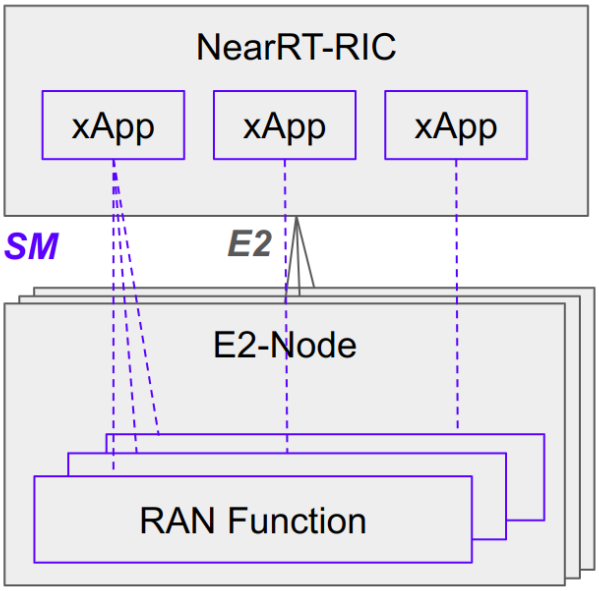

Figure 5: xApp Communication with E2-Nodes.

- RAN functions are the internal functions within an E2-Node responsible for managing UEs and Cells (e.g., MAC scheduler, stats of RLC, PDCP and RRC layers)
- Service Model (SM) serves as a framework for storing collected data and control information, and it is used to communicate between xApp and RAN functions. There are several SMs existing: (1) Standardized SMs from O-RAN: Key Performance Measurement (KPM), Radio Control (RC), and Cell Configuration and Control (CCC), and (2) Customized SMs from FlexRIC: MAC, RLC, PDCP, GTP, Slice Control (SC), and Traffic Control (TC)

3 Open RAN Deployment

This experiment is to deploy a 5G Standalone (SA) network using OpenAirInterface (OAI) [2] RF Simulator gNB and OAI minimal 5GC, but this time using the F1 Split architecture. We also deploy FlexRIC as the Near-RT RIC with a monitoring xApp. You need to edit the file `open-ran.yaml` for this section. Similar to the previous labs, apply the changes to the PLMN.

3.1 Deployment and Testing

Deploy the file using the command `cli install network open-ran.yaml`. Check for the status of the deployment using the command `cli observe` and make sure all the Elements other than the UEs are in the `1/1 Y` state.

Open two terminals and on each apply the filters as below and open wireshark for the CU and DU respectively.

[2] https://openairinterface.org/

```
$ udp_ports="2123 or 2152 or 9999"
$ sctp_ports="38412 or 38472 or 36421"
$ filter="(sctp port $sctp_ports) or (udp port $udp_ports)"
$ cli extract pcap {element} -- "$filter" | wireshark -k -i -
```

You should replace the `{element}` with the name of the Element name from the observe command output.

Then use the `t-dumper` command to start the CU and DU. For each of them **separately** run the following command:

```
$ cli cic {element} run -- t-dumper
```

Use `Ctrl+C` to stop the command, after you see the line is printed.

Note

Make sure you run first the CU and then the DU. Due to a bug in OAI code for multiple UEs, you have around 2 minutes to run all the previous commands, otherwise the UEs would not be able to connect to the DU due to a race condition.

Questions

1. How do we specify the deployment type of the gNB in the `open-ran.yaml` file?
2. How do we specify the CU and DU to be connected to the NearRT-RIC?
3. Map the architecture of the network to the O-RAN architecture by mentioning the IP addresses of each component.

Important Note

The order between the NGAP and F1AP or F1AP and MAC messages might be wrong in the PCAP files just by a few microseconds needed to capture and process the packets. However, the order between the messages in the same interface is always correct.

3.2 F1 Split Questions

Use the PCAP files for the CU and DU to answer the following questions. Choose any MAC-NR message and expand the Context field and right-click on the Slot, System Frame Number, and Direction, then Apply them as a column.

1. In which message is the cell information transmitted from the DU to CU, and what is the NR cell identity?
2. In which message CU decides on the activated cells?
3. Draw a message sequence chart including the UE, CU, and DU for the random access procedure (messages 1 to 5). You could use https://plantuml.com/ for drawing the chart.
4. What is the difference in the random access procedure between the F1 split and without F1 split?

5. Use the protocol stack in the figure 2 to find the pair of the messages for the CU and DU in the PCAP file. For example, for the RRC Setup Response message in CU, there should be a corresponding F1AP message carrying the RRC message from CU to DU. These two messages are using different protocol stacks (one F1AP and the other NR-RRC), but they are paired with each other. Find the message numbers and the corresponding protocol stack layer for each pair.

6. Use the messages in the CU or DU PCAP file to fill the table 1 for each UE, while mentioning the message ID, protocol stack layer, and node type (CU or DU).

7. Since the DU does not process RRC messages, which F1AP message notifies the DU that the RRC connection setup is completed?

8. In which messages the SRB1, SRB2, and DRB1 are setup and through which logical channels?

9. What is the relation between the slot number, direction, and the TDD pattern?

10. What is the maximum System Frame Number?

11. How much time is passed until the System Frame Number resets?

12. Describe the possible handover scenarios in the Open RAN architecture by referencing figure 1.

UE	RA-RNTI	C-RNTI	gNB-CU UE-ID	gNB-DU UE-ID	AMF UE-ID
1					
2					

Table 1: F1 Split Messages

Some useful links for the handover scenarios:

- https://www.techplayon.com/5g-sa-handover-inter-gnb-du-and-intra-gnb-cu-handover/
- https://www.techplayon.com/5g-sa-inter-gnb-hanodver-xn-handover/
- https://www.techplayon.com/5g-sa-inter-gnb-handover-n2-or-ngap-handover/

3.3 GTP and RLC Questions

1. Starting from the NGAP PDU session setup request message, draw a message sequence chart to explain the steps to setup a PDU session including the UE, CU, DU, AMF, and UPF.

2. For the PDU session for each UE, find the 5QI value (written as `fiveQI` in Wireshark) of its QoS flows, the PDU session type, and the Allocation and Retention Priority (ARP) level.

3. What is the PDU session Aggregate Maximum Bit Rate (AMBR) for each PDU session?

4. What is the AMBR for each UE?

5. Why there are two separate AMBR defined (even though the values are equal in this case)?

6. Fill the table 2 by the values from the messages above for each of the two UEs.

7. Explain in which order the TEIDs are allocated for each UE.

UE	UPF TEID	CU to UPF TEID	CU to DU TEID	DU to CU TEID	Assigned IP
1					
2					

Table 2: GTP Tunnel Endpoints

Now use the command `cli test rtt ue1 -- -s 2000 12.1.1.1` and then in the DU Wireshark, use the filter `rlc-nr.am.si != 0x0` to find the RLC segmented packets.

8. How many RLC segments are created for each ICMP packet?

9. How many fragmented IP packets are created for each ICMP message?

10. Now change the packet size to 1200 bytes and answer the questions again.

11. What is different from between the two cases?

12. Find the maximum size for the ICMP packets (the parameter `-s`) that does not trigger segmentation in RLC.

3.4 Open RAN Questions

Having a malformed packet for E2 setup request is normal. Your Wireshark version cannot dissect E2AP v2 messages used by FlexRIC.

1. How many E2-Node(s) connected with the NearRT-RIC?

2. What E2-Node(s) information are included in the E2 Setup Request?

3. We have subscribed to KPM, PDCP, and MAC service models. Explain which E2-Node(s) are subscribed for each service model and measure the RIC indication interval.

4. What are the supported RAN functions in a E2-Node (please explain each E2-Node respectively)?

5. Draw the message sequence chart (use MSCGen again) between NearRT-RIC and a E2-Node (Start from the E2 Setup Request).

6. What is the RIC indication type for each service model?

7. Assuming the value of the inter-arrival for the MAC service model, how many records from the database should be recorded to cover the last 45 seconds of both UEs metrics? *Hint: You could use the filter* `e2ap.RANfunctionID == {value}` *to filter the records for each Service Model.*