



ITU-ML5G-PS-014: Build-a-thon(PoC)

# Autonomous Resource Allocation for Emergency Network Slice

---

TEAM: AUTOMATO

MEHMET KARACA & DORUK TAYLI & ÖZGE SİMAY DEMİRCİ

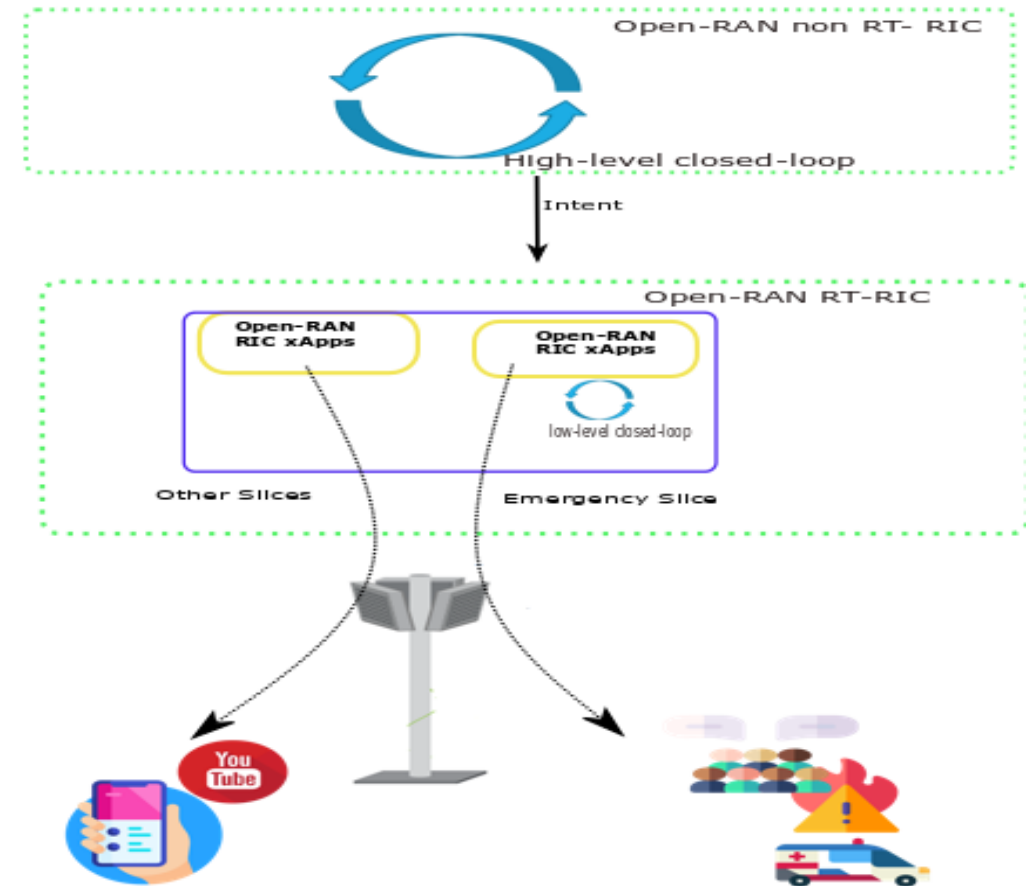
30 NOV 2021

HOST: ITU FOCUS GROUP AUTONOMOUS NETWORKS (FG-AN)

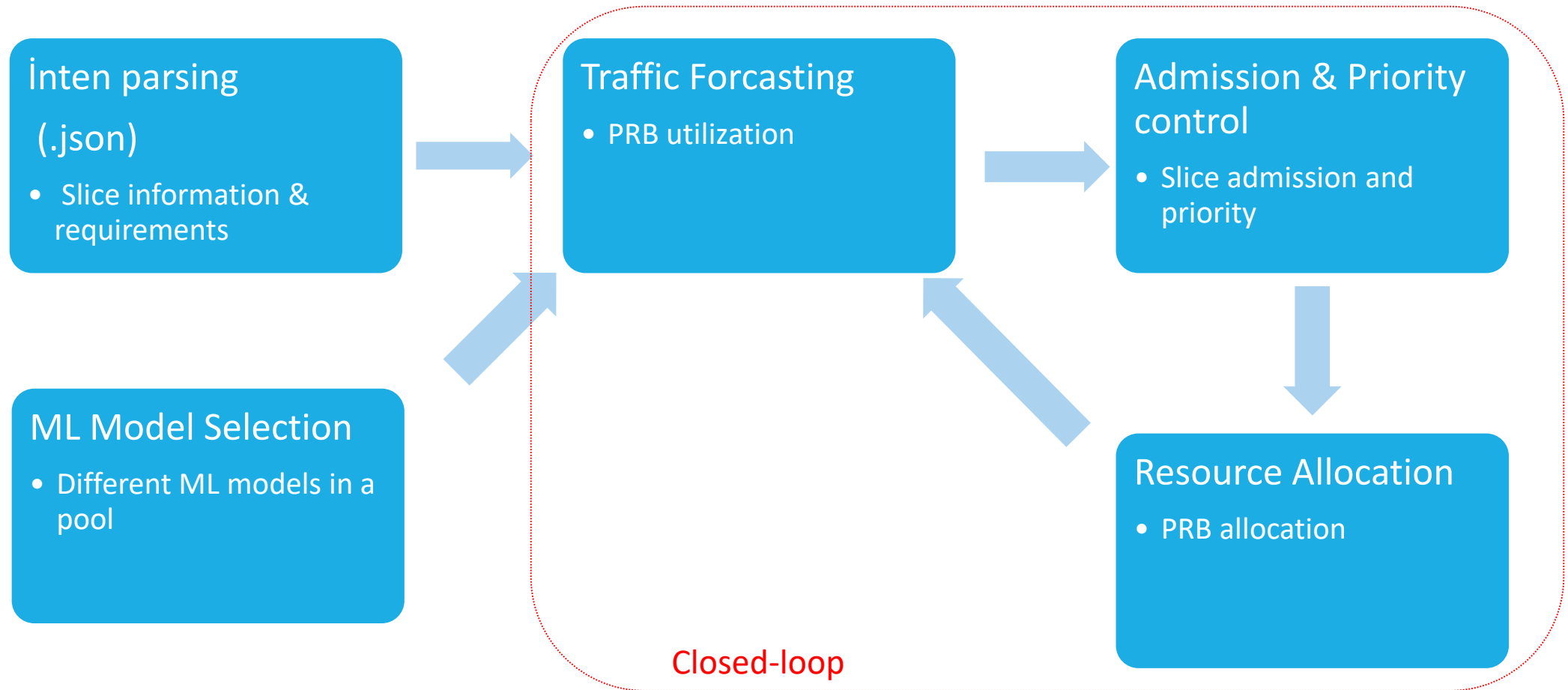
# Problem Definition

Make resource allocation decision for **emergency slice in an autonomous way**:

- ❑ Get & parse high-level intent
  - Requirements for ES
  - Information regarding data, ML models
- ❑ Make traffic analysis
- ❑ Admission & priority control
- ❑ Make resource allocation

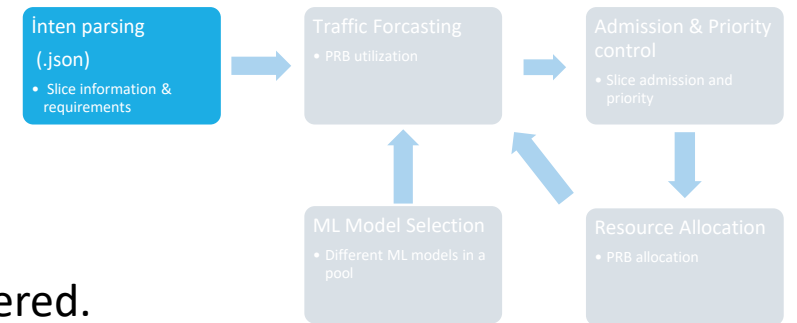


# Building Blocks



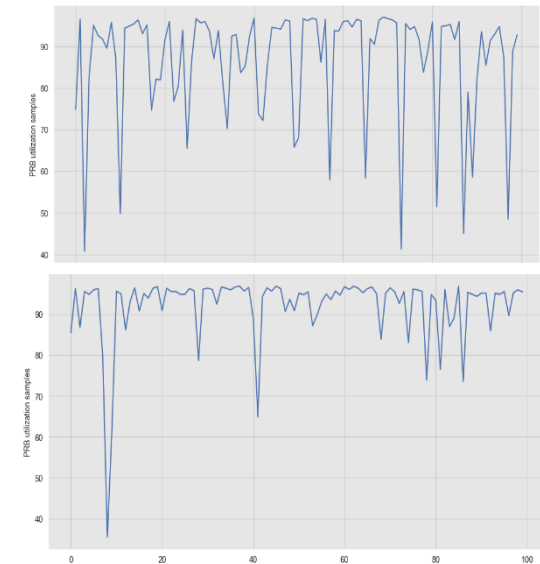
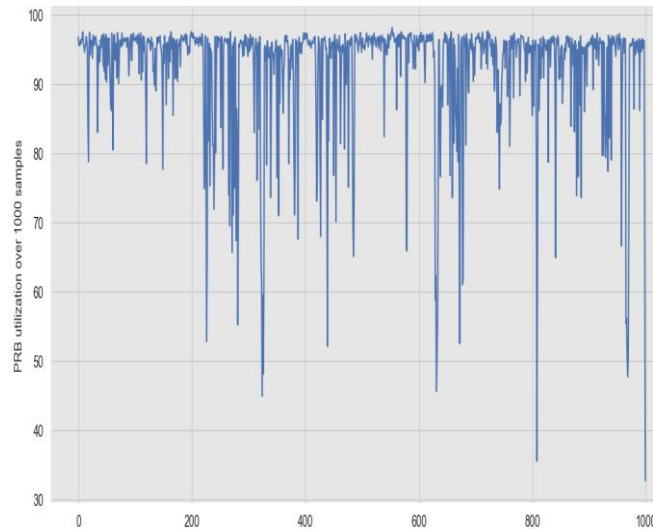
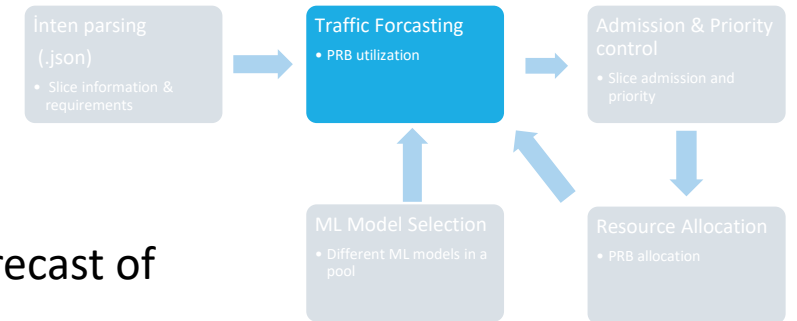
# Intent Parsing

- ❑ We get intent from higher-loop (Open-RAN NonRT-RIC).
- ❑ Indicates if there is an emergency case and monitoring xApp is triggered.
  - Information about where data and ML models are.



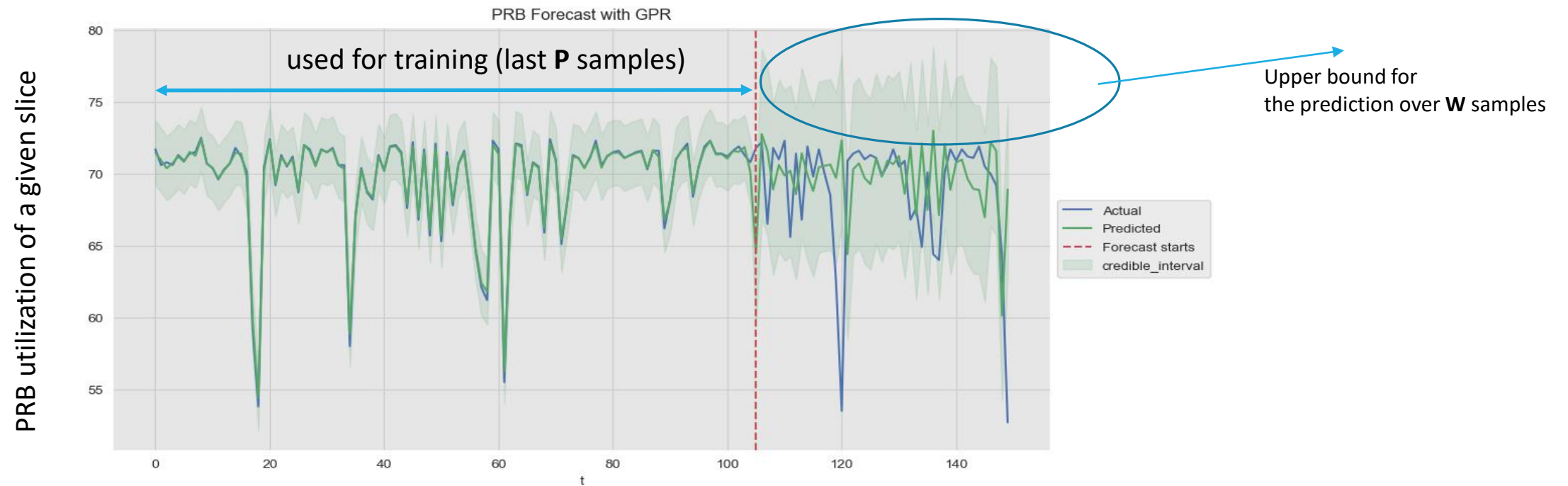
# Traffic Forecasting

- We monitor RAN resource utilization
  - Checks for available PRBs for emergency slice
  - We apply Gaussian Process Regression (GPR) for time-series forecast of PRB utilization



# Traffic Forecasting (cont.)

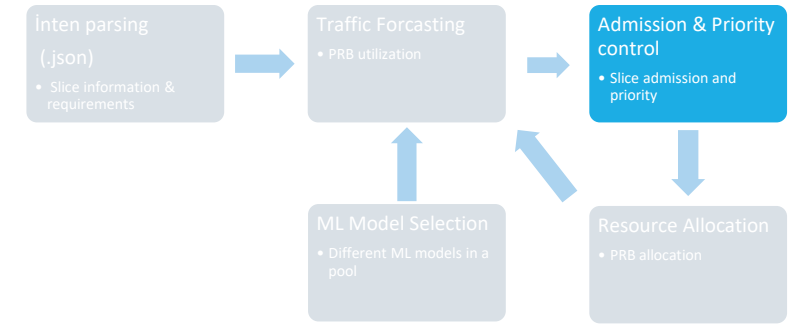
- ❑ We develop this building block as a **RIC xApp**. This is our first xApp for this problem.
- ❑ Real data (PRB utilization) taken from paper [1].



[1] Vaclav Raida , Philipp Svoboda, Markus Rupp : “Real World Performance of LTE Downlink in a Static Dense Urban Scenario - An Open Dataset “ , IEEE GLOBECOM 2020.

# Priority Control

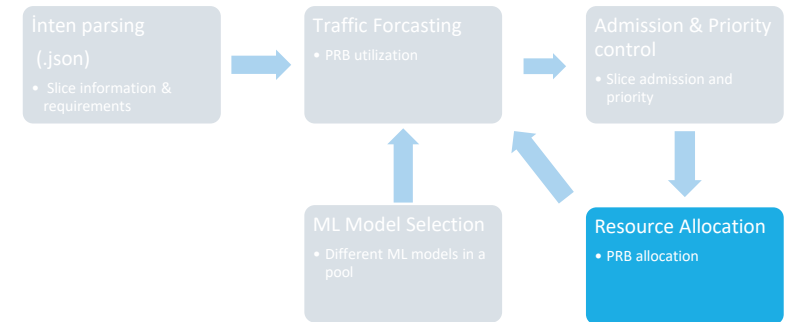
- ❑ RAN is shared with other possible slices that have certain number of PRBs dedicated to them. How much resource should we allocate to emergency slice (ES)?
- ❑ Level of emergency
  - Protect the existing slices first (ES is the second priority)
    - Algorithm 1 (ALG 1)
  - Always give the first priority to ES
    - Algorithm 2 (ALG 2)



# Resource Allocation

---

- ❑ ALG 1 and ALG 2 decide on resource allocation
  - How many PRB should we allocate for ES?
  
- ❑ Resource Allocation block is implemented as **RIC xApp**. This is our **second RIC xApp**.





# ALG 1

**IDEA:** Amount of PRBs allocated to other slices are not always used (underutilization). Predict unused PRBs of other slices (leftover from other network slices) and allocate them to ES.

**For each other slice n**

Step 1: Train GPR with the latest P training data

Step 2: Forecast PRB utilization over the next W samples with GPR  $\rightarrow U_n$

Step 3: Calculate maximum possible PRB utilization  $\rightarrow C_n = U_n + o_n$

Step 4: Calculate forecasted PRB usage over next W samples  $\rightarrow B_n = T_n C_n$

**End**

Step 5: Calculate available PRBs for Emergency Slice  $\rightarrow P_{ES} = T - \sum_{n=1}^N T_n C_n$

Step 6: Allocate PRBs to ES  $\rightarrow P_{ES}$

Resource  
allocation  
(ALG 1)

Example: Two slices

Total system PRBs = 100

Amount of PRBs allocated to first slice is 40,  
utilization is %80 (predicted)  $\rightarrow$  **8** PRBs unused

Amount of PRBs allocated to second slice is 60,  
utilization is %90 (predicted)  $\rightarrow$  **6** PRBs unused

PRBs allocated to emergency slice is **8 + 6 = 14**  
PRBs

# ALG 2

**IDEA:**Emergency slice needs **E** amount of PRBs (fixed). Allocate the available PRBs to ES first. If it is not enough borrow PRBs from other slices. Aim is to always give **E** PRBs to ES. ES has the first priority.

Resource  
allocation  
(ALG 2)

Solve an optimization problem:

$$\begin{aligned} \min \quad & \sum_{t=1}^W \sum_n^N \max\{0, x_n(t) - (T_n - y_n(t))\} \\ \text{s.t.} \quad & 0 \leq y_n(t) \leq T_n \quad \forall n \\ & \sum_n^N y_n(t) \geq E \end{aligned}$$

PRB needed for slice n at time t

Total PRBs given to slice n

Number of PRB taken from slice n at time t to be used for emergency slice

We damage slice n this amount by taking PRBs from it

Guarantee that emergency slice gets enough PRBs

**Transform this problem to a solvable integer problem**

[2] X. Foukas et al., "Orion: RAN Slicing for a Flexible and Cost-Effective Multi-Service Mobile Network Architecture," in ACM MobiCom, 2017.

[3] Armin Okic, Lanfranco Zanzi, Vincenzo Sciancalepore, Alessandro Redondi, Xavier Costa-Pérez, "π-ROAD: a Learn-as-You-Go Framework for On-Demand Emergency Slices in V2X Scenarios", IEEE INFOCOM, 2020.

# ALG 2 (cont.)

**IDEA:** Emergency slice needs **E** amount of PRBs (fixed). Allocate the available PRBs to ES first. If it is not enough borrow PRBs from other slices. Aim is to always give **E** PRBs to ES. ES has the first priority.

Resource  
allocation  
(ALG 2)

Solve an integer programming using auxiliary variable  $u_n$ :

$$\min \sum_{t=1}^W \sum_n^N u_n(t)$$

$$\tilde{x}_n(t) + o_n(t) - (T_n - y_n(t)) \leq u_n(t)$$

$$0 \leq y_n(t) \leq T_n \quad \forall n$$

$$\sum_n^N y_n(t) \geq E$$

$$u_n(t) \geq 0$$

$$x_n(t) = \tilde{x}_n(t) + o_n(t)$$

Actual PRB usage at  
time t in future.  
This cannot be  
known in advance

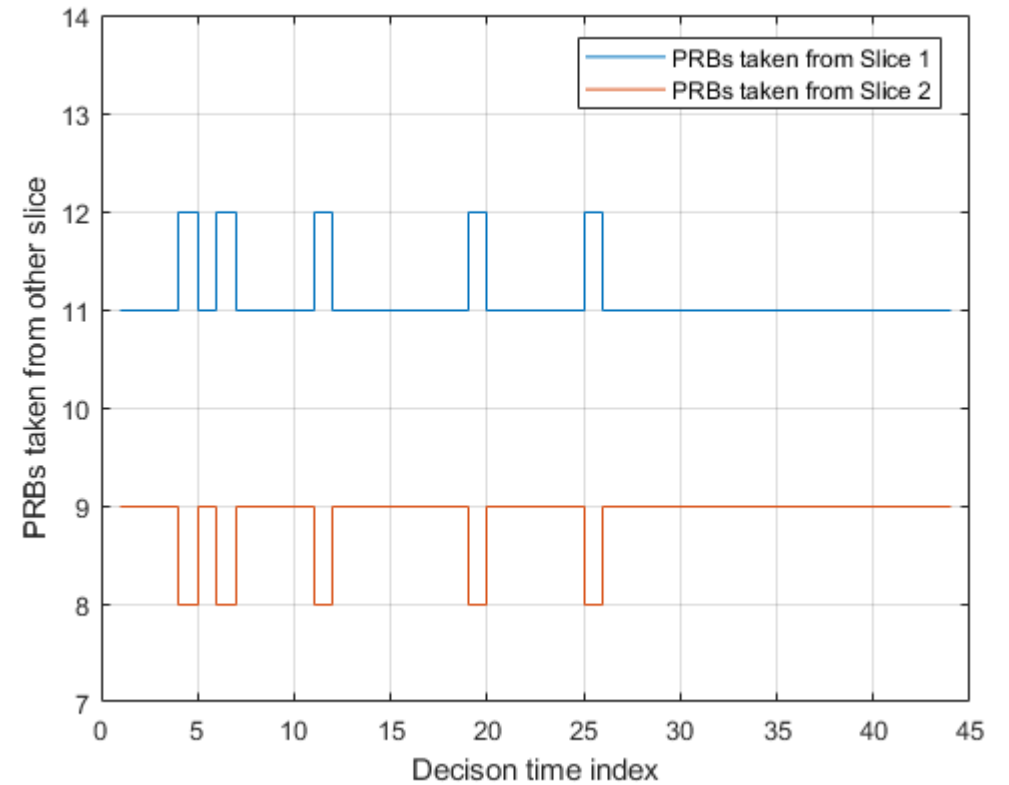
Estimated PRB usage  
with GPR

Estimation error.  
Upper bound can be  
used provided by  
GPR.

**We have implemented ALG1 & ALG2 and next we will show a demo for ALG2**

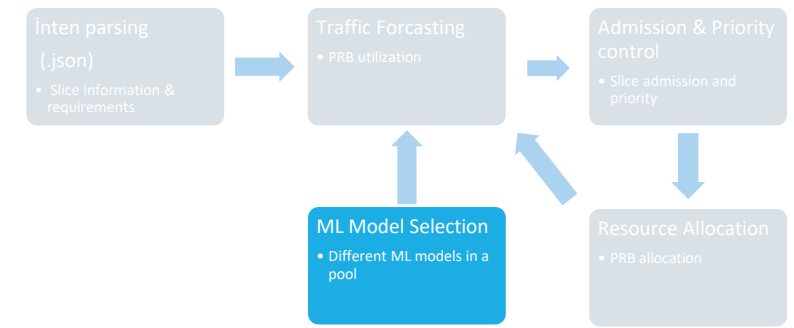
# Experiment: resource allocation

- Two other slices:
  - T40 and 60 PRBs allocated these two slices
  - he network has 100 PRBs (ES should borrow from others)
- Emergence slice needs **E = 20** PRBs
- Result from the Figure: at every decision-time, sum of these PRBs is always 20, which ES needs.

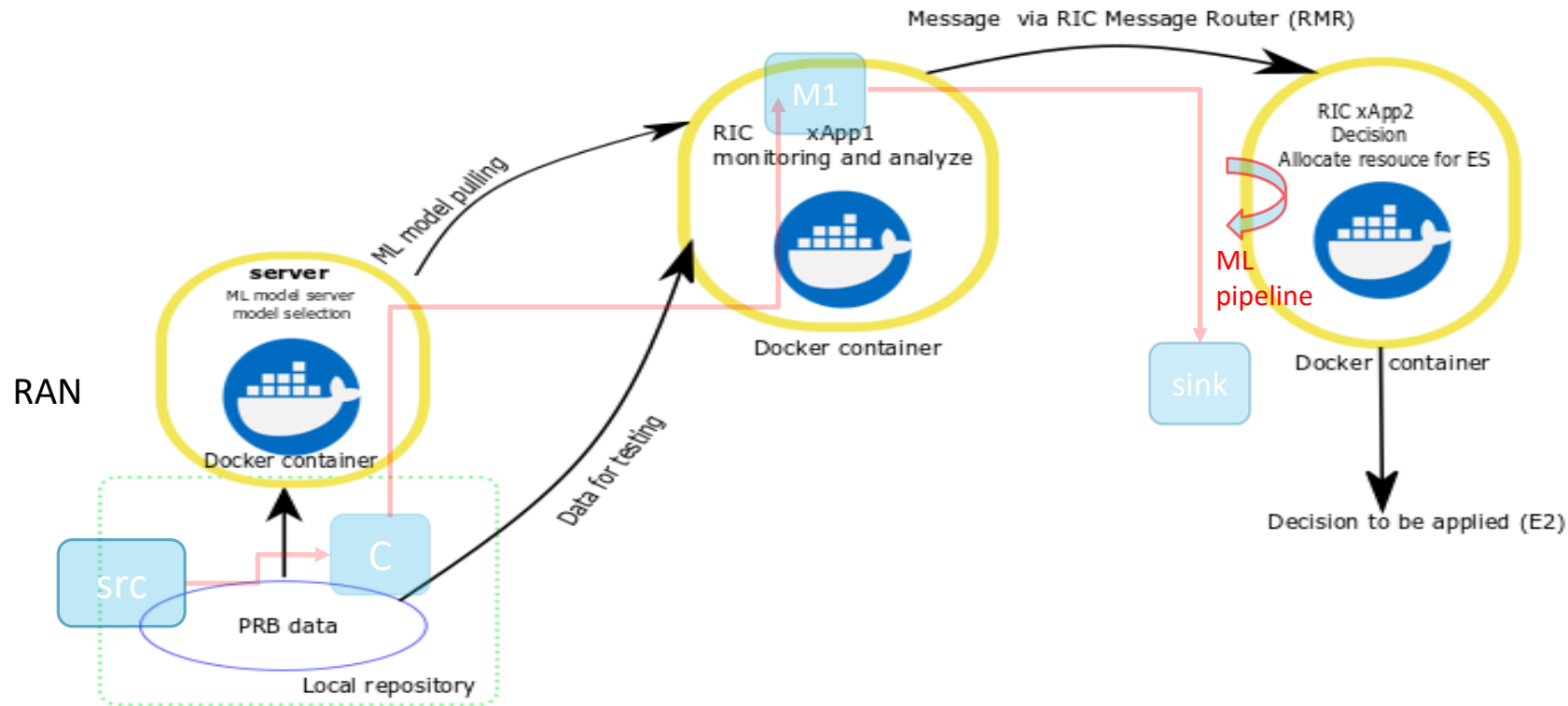


# ML model selection for forecasting

- ❑ Different ML models with different performance, complexity and cost can be available in a pool.
- ❑ We implement this scheme as a microservice (docker).
  - Webserver stores different ML model
  - Possible to fetch a different ML model when forecasting



# Our implementation



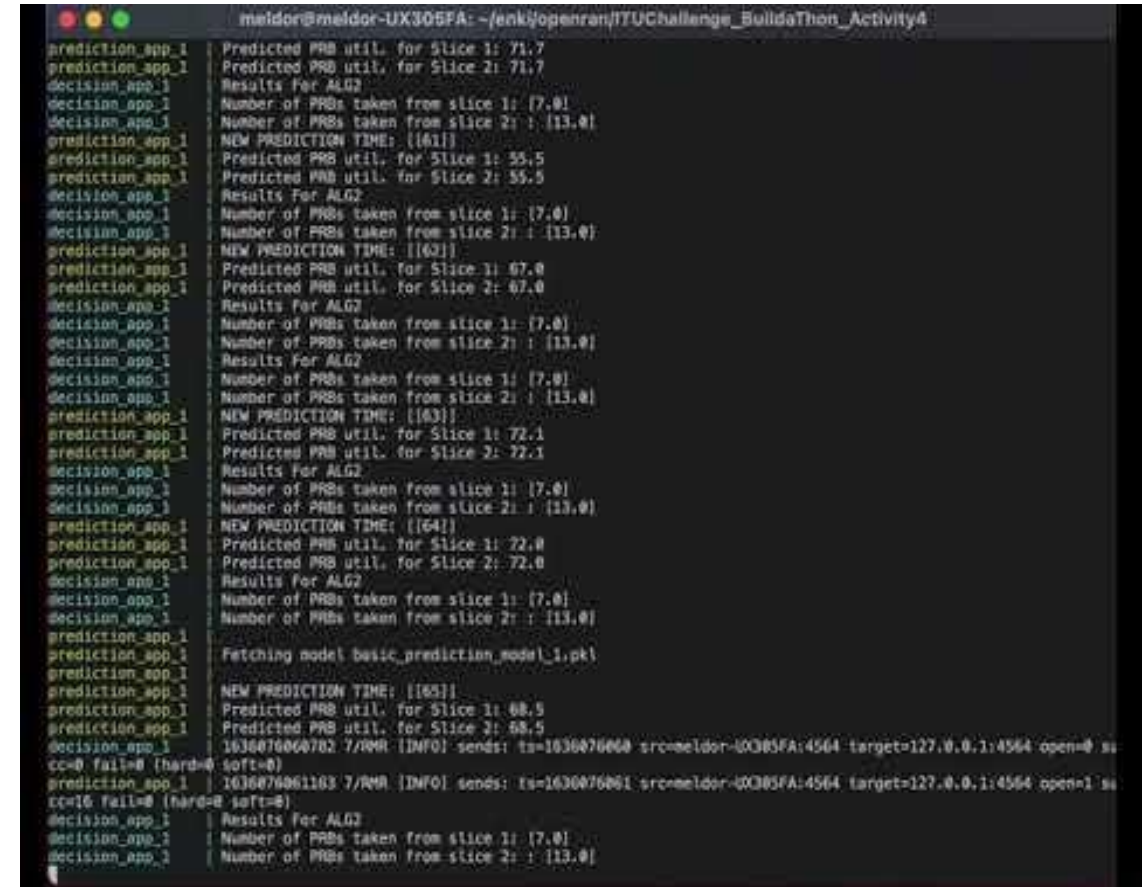
- ❑ Implemented RIC xApp1, xApp2 and server as a microservice.
- ❑ These xApps can communicate through a RIC (O-RAN messaging) message.
  - e.g., xApp1 can send prediction to xApp2 over the network

# PoC Demo

We have a quick demo:

- ❑ Ubuntu 20.04 Works fine
- ❑ Need docker and docker compose

<https://drive.google.com/file/d/1ouqhovouHlZiYOotnKqFHylB8JUDdAFIF/view?usp=sharing>

A terminal window titled 'maldor@maldor-UX305FA: ~/enk/openran/ITUChallenge\_BuildaThon\_Activity4' displays the output of a PoC demo. The output shows a series of predictions and results for two slices (Slice 1 and Slice 2) across multiple iterations. The predicted PRB util. for Slice 1 is consistently 71.7, and for Slice 2 is 71.7. The number of PRBs taken from slice 1 is 7.0, and from slice 2 is 113.0. The results for ALG2 are also shown. The terminal output is as follows:

```
maldor@maldor-UX305FA: ~/enk/openran/ITUChallenge_BuildaThon_Activity4
prediction_app_1 Predicted PRB util. for Slice 1: 71.7
prediction_app_1 Predicted PRB util. for Slice 2: 71.7
decision_app_1 Results For ALG2
decision_app_1 Number of PRBs taken from slice 1: [7.0]
decision_app_1 Number of PRBs taken from slice 2: [113.0]
prediction_app_1 NEW PREDICTION TIME: [[61]]
prediction_app_1 Predicted PRB util. for Slice 1: 55.5
prediction_app_1 Predicted PRB util. for Slice 2: 55.5
decision_app_1 Results For ALG2
decision_app_1 Number of PRBs taken from slice 1: [7.0]
decision_app_1 Number of PRBs taken from slice 2: [113.0]
prediction_app_1 NEW PREDICTION TIME: [[62]]
prediction_app_1 Predicted PRB util. for Slice 1: 67.0
prediction_app_1 Predicted PRB util. for Slice 2: 67.0
decision_app_1 Results For ALG2
decision_app_1 Number of PRBs taken from slice 1: [7.0]
decision_app_1 Number of PRBs taken from slice 2: [113.0]
prediction_app_1 NEW PREDICTION TIME: [[63]]
prediction_app_1 Predicted PRB util. for Slice 1: 72.1
prediction_app_1 Predicted PRB util. for Slice 2: 72.1
decision_app_1 Results For ALG2
decision_app_1 Number of PRBs taken from slice 1: [7.0]
decision_app_1 Number of PRBs taken from slice 2: [113.0]
prediction_app_1 NEW PREDICTION TIME: [[64]]
prediction_app_1 Predicted PRB util. for Slice 1: 72.0
prediction_app_1 Predicted PRB util. for Slice 2: 72.0
decision_app_1 Results For ALG2
decision_app_1 Number of PRBs taken from slice 1: [7.0]
decision_app_1 Number of PRBs taken from slice 2: [113.0]
prediction_app_1 Fetching model basic_prediction_model_1.pkl
prediction_app_1 NEW PREDICTION TIME: [[65]]
prediction_app_1 Predicted PRB util. for Slice 1: 88.5
prediction_app_1 Predicted PRB util. for Slice 2: 88.5
decision_app_1 1636076064782 7/7/2021 [INFO] send: ts=1636076064 src=maldor-UX305FA:4564 target=127.0.0.1:4564 open=0 ss
cc=0 fail=0 (hard=0 soft=0)
prediction_app_1 1636076064783 7/7/2021 [INFO] send: ts=1636076064 src=maldor-UX305FA:4564 target=127.0.0.1:4564 open=1 ss
cc=16 fail=0 (hard=0 soft=0)
decision_app_1 Results For ALG2
decision_app_1 Number of PRBs taken from slice 1: [7.0]
decision_app_1 Number of PRBs taken from slice 2: [113.0]
```

Check for details: <https://github.com/ITU-AI-ML-in-5G-Challenge/ITU-ML5G-PS-014-Build-a-thon-PoC---Team-AUTOMATO>

# Conclusion and Future Works

---

- ❑ A PoC for autonomous slice management
- ❑ Resource allocation handled with an autonomous closed-loop
- ❑ Machine learning with linear optimization
- ❑ Flexible and modular O-RAN RIC xApps design

## Future Works:

- ❑ Integration of all xApps to testbed, real O-RAN infrastructure
- ❑ Other optimizations & improvements
- ❑ Dynamic estimation of emergency slice



---

# Thank you for listening!

## Q & A

mehmet.karaca@tedu.edu.tr  
doruktayli@gmail.com  
osimay.demirci@tedu.edu.tr