CUSTOMIZED XAPP FOR OPENRAN COMPLIANT DEPLOYMENT: A 5G STANDALONE APPROACH

Gabriel-Mihai OLTEAN, Andrei-Bogdan RUS, Daniel ZINCA, Virgil DOBROTA
Communications Department, Technical University of Cluj-Napoca, Romania
gabriel.oltean09@gmail.com ; Bogdan.Rus@com.utcluj.ro; Daniel.Zinca@com.utcluj.ro
Corresponding author: Virgil Dobrota (e-mail: Virgil.Dobrota@com.utcluj.ro)

Abstract: The paper presents an OpenRAN compliant deployment in 5G Standalone, based on the srsRAN Project version 24.10.1 (upgraded from 24.10 due to several bugs in configuration), Open5GS v2.7.2 acting as Core Network and O-RAN SC RIC (O-RAN Software Community Near-Real-time RAN Intelligent Controller). Lack of support from vendors and suppliers led to issues and slowdowns in deploying a more complex testbed. However, the paper manages to demonstrate the compatibility between the equipment involved (such as Milesight 5G AI 4X/12X Pro Bullet Plus CCTV Camera, Nokia FastMile 5G Receiver 5G26-A, Ettus Research B200 USRP) and the software mentioned above. This was acquired by writing a customized xApp (i.e., application running in Near-Real Time RIC), containing a custom logic function that reads the throughput data every second, comparing it to a moving threshold value and increasing/ decreasing the maximum number of PRBs (Physical Resource Blocks) that can be allocated to a User Equipment in correlation to how much data it requests.

Keywords: *5G Standalone, Open RAN, xApp.*

I. INTRODUCTION

The 5G networks are currently deployed, but each operator has implemented a specific version of them. However, the costs are quite high, and solutions for reducing them are desperately needed. One of these avenues is represented by the OpenRAN concept, governed by the O-RAN ALLIANCE [1]. This is a consortium of network operators, research laboratories, and network equipment vendors, working together to "open" the Radio Access Network. Finally, it will be able to break free from the tight integration of hardware and software components that have previously defined the RAN. The concept is to split into multiple physical and logical components, to define standard interfaces between them, and to get different vendors to implement this into their devices. This has been the mission of the O-RAN ALLIANCE since its creation by the five founding members: AT&T, China Mobile, NTT Docomo, Orange, and Deutsche Telekom AG.

Open-RAN implementations are here, and in later years more papers have surfaced testing various scenarios and experimenting with innovative AI/ML technologies for improving the security of the RAN, the efficiency of resource allocation of the radio spectrum, or the integration between devices from different vendors while maintaining the expected performance of the network. Within its architecture, one can find the RICs (RAN Intelligent Controllers), which are components capable of taking intelligent, closed-loop action within the RAN.

While several solutions change the allocation policy by taking into account the overall number of users and their type, this paper focuses on supplying the best possible connection with the fewest resources for a certain user, with plans to expand the algorithm for multiple users and eventually to leverage AI/ML algorithms to determine the dominant type of traffic for each user and to modify the resource allocation strategy in order to account for the known requirements of different types of traffic.

The remainder of the paper is structured as follows. Section II presents the related work, from machine-learning approaches up to cloud resource management. Section III is focused on implementation, the deployment process and the xApp. Next section presents the experimental results, and it shows the workings of the xApp and its performance. Finally, Section V concludes the paper and outlines future work such as redesigning this xApp as an ML powered application.

II. RELATED WORK

The goal of paper [2] was to show how to extract the maximum power out of the near-RT (near-Real Time) and non-RT RICs, leveraging the available AI/ ML (Artificial Intelligence/Machine Learning) frameworks. The paper first goes on to define the old RAN approach as a vendor-locked black box and to expand on the advantages of the O-RAN architecture while taking into consideration the business requirements for new applications of the 5G and B5G (Beyond 5G) technologies. Then it goes on to explain in depth the specific functions of the RICs and how they can be leveraged in the AI/ML context. In brief, the Near-RT RIC hosts one or more xApps that use the E2 interface to collect information and to determine actions regarding one or more specific connections between the UE and the RAN, while the non-RT RIC can use the A1 interface to push declarative policies to the Near-RT RIC. In terms of ML, the paper defines multiple scenarios in which a given ML application can be deployed either in the Near-RT RIC if the focus is on quick (sub-1s) decisions or in the non-RT RIC if the focus is on longer trends and reconfigurations. Regarding the ML/AI part of the paper, the authors have identified multiple deployment options focusing on different components with various advantages and disadvantages. One such option talks about having continuous operation, model management, and data preparation for training and the training itself live in the non-RT RIC. On the other hand, the near-RT RIC hosts the modules responsible for data collection, data preparation, and finally inference. This option provides great flexibility and agility in terms of the ML components. The last part of

Manuscript received November 3, 2025; revised December 14, 2025

the paper presents in detail a subset of use cases defined by the O-RAN ALLIANCE. It was focused on SLA assurance, resource allocation, traffic steering, and energy efficiency. Moreover, it talks about the challenges that researchers have faced whilst developing new applications for the RAN. As it stands, the biggest challenges reside with the conflict between new and existing required apps, with the compatibility between the existing RICs and the newly developed xApps, with the lack of a Real-Time RIC component which does not yet exist.

One of these main challenges was to create an xApp. By the time of writing this paper, the lack of proper support and documentation makes the current learning curve of this technology stack quite steep, as described by [3]. Their research into Beam Mobility Management was hindered by ambiguities in either the O-RAN ALLIANCE specification on "AI/ML-assisted Beam Selection Optimization", which leaves room for questions regarding the Location Information, the ML modules and the E2 interface. Moreover, when discussing the Signaling Storm Detection app, the authors found other ambiguities regarding the resolution of the TA, the Non-RT RIC architectures in terms of processing of EI, and regarding the E2 interface policy service.

There are relatively few publications that approach the subject of dynamic resource allocation of radio resources in the 5G spectrum using xApps. One that can have found of interest is [4], which takes an ML approach to this issue. It proposes four distinct resource allocation policies that split the number of available PRBs between Voice and Mobile Broadband traffic. This is either in equal proportions, or favoring one over the other, by using dedicated resources for each type of traffic. The goal of the algorithm is to consider the network conditions, the number of users, and their type, and with this info to be able to choose the optimum resource allocation policy. Each of the three tested classifiers seemed to show a good performance in selecting the optimum resource allocation policy, i.e., their accuracy on the validation dataset was more than 85%. Out of the three algorithms, the Random Forest Classifier shows the smallest training time and has been deemed by the authors of [4] as the best candidate to be the focus of their study.

The data recovered after implementing the xApp and running simulations of the model in a virtual scenario has yielded that within macro cells, the MBB Priority Allocation policy has been rarely used, while the same remark can be made in the case of small cells about the Voice Priority Allocation Policy. The paper concludes that the integration of AI/ML capabilities within the O-RAN brings considerable advantages; moreover, the dynamic resource allocation technique used leads to fewer outage events, resulting in an optimization of the used resources.

In contrast to [4], this paper has expanded the subject from the virtual world into the real world. Thus, the complexity of the dynamic allocation policy may be reduced, but the overall complexity of the testbed is greatly increased as it had to consider the deployment of the end-to-end 5G standalone network, the compatibility between the RAN the Core and the UEs used various sources of interference and the predictability of real traffic requirements and scenarios. To this end, more time has been spent getting all the components of the system to communicate with each other than creating the dynamic allocation algorithm. Even though both [4] and this paper deal with dynamic resource allocation in 5G networks, they take two different approaches; the algorithm used in this paper is a classical one, while [4] uses AI/ML algorithms.

III. IMPLEMENTATION

One of the most important elements in understanding the system described by this paper is the interpretation of the above-described theory into a coherent test bench. Thus, Figure 1 presents the approach of building a 5G SA O-RAN compliant network, based on preliminary work carried out by the authors in [8]. This diagram represents the first step in building this project and is a high-level representation of the system and the hardware devices to be used.

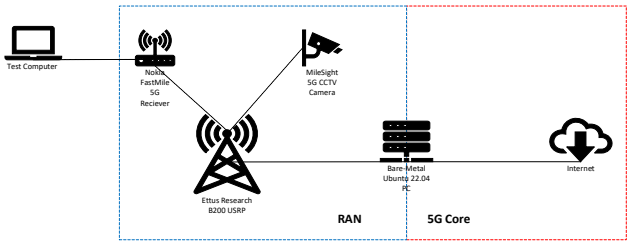

Figure 1. High level testbed diagram

Four major hardware components were used: (1) Server PC running Ubuntu 22.04 on bare metal; (2) Ettus Research B200 USRP; (3) Nokia FastMile 5G Receiver 5G26-A; (4) Milesight 5G AI 4X/12X Pro Bullet Plus CCTV Camera. Each of them had specific capabilities that have influenced the way this deployment was designed. It must be brought to the reader's attention that using other devices than those specified above may lead to incompatibility with the settings that will be provided in what follows, or even with other software used.

The Bare-Metal Ubuntu 22.04 PC sits in the middle of everything, being a part of both the 5G RAN and the 5G Core. This is because it ran two main applications: the srsRAN Project, which implements the 5G RAN, and the Open5GS, which implements the 5G Core. As presented in the theoretical part, the RAN can be "split" at different points to create systems with certain advantages and disadvantages depending on the specific split point of the RAN. In this instance, due to hardware limitations, using a B200 USRP and not a full-fledged RU, a "split option 8" configuration was implemented, where the CU and DU are running in software on the PC and the RU is represented by the USRP (Universal Software Radio Peripheral).

The Nokia FastMile 5G Receiver and the MileSight 5G CCTV Camera are used as test UEs, one being an end device (MileSight 5G CCTV Camera), and one being used as a SOHO router allowing for more devices to be connected behind it, in this case a computer that now has access to the internet via a 5G connection. On the core side of the diagram, the Open5GS cluster of applications makes sure that the devices in the RAN have access to the internet, and some routes set on the PC make sure that the traffic can exit the 5G Core and carry on to their destination on the Internet.

The Nokia router was the first device tested, as it was decided that it would be easier to test the internet connection by using a laptop attached to the router and confirm that there is indeed Internet connectivity [5]. This device is an outdoor 5G receiver boasting high-gain

(10dBi) antennas arranged in such a way as to cover a 180° field of view. It can do carrier aggregation up to 300MHz and is compliant with 5G SA systems up to 3GPP release 16. On the LAN side, it has a 2.5G POE port. It comes with a PoE adaptor in a box to be able to power the devices and to connect a computer to the LAN port. By default, it has a DHCP server active serving IP addresses from the subnet 192.168.1.0/24, where the Nokia router acting as gateway can be accessed at 192.168.1.1.

The high-end MileSight CCTV camera supports resolutions of up to 4K (3840x2160) and a framerate of up to 30 fps. It can encode the video using either H.264 or H.265 standard in either CBR (Constant Bit Rate) or VBR (Variable Bit Rate) modes. The CBR mode encodes the media at a fixed bit rate regardless of the content's complexity allowing for predictable bandwidth usage whilst potentially sacrificing image quality and the VBR mode adjusts the bitrate based on the complexity of the media to be able to achieve better video quality with the drawback being that the bandwidth usage is much less predictable. When talking about the connectivity side of the camera, it has a 10/100 Ethernet port, which supports PoE to power the camera and allows for the direct connection to the camera for settings. The camera has, by default, the IP address 192.168.5.190/24 and it can be accessed by a computer connected to the same subnet. The interface is complex, and the network tab has both a basic and an advanced mode. In the basic mode, one will find the settings for the Cellular interface. The camera supports both the 5G SA and the 5G NSA mode in the bands n1, n2, n3, n5, n7, n8, n12, n20, n28, n41, n66, n71, n77, n78, n79. According to [6], the camera had a maximum wireless transmission speed of 3.3 Gbps in the downlink direction and 250 Mbps in the uplink direction, using 5G SA mode.

For both UEs sysmoISIM-SJA5 SIM cards were used which were programmed using an "Axagon Smart Card PocketReader" and the "pySim" python library.

The Open5GS core setup is not custom in any way and can be followed from the manufacturer website, to be noted that it is necessary for it to be fully installed with all components running including the GUI, IP forwarding rules and NAT rules, to ensure communication of the client devices with the rest of the network. Other specific configuration steps are within the GUI, where one must register the SIM cards, as well as the APN names and settings that it expects the UEs to use.

Regarding the 5G RAN there are multiple methods to deploy it, one of them is being presented in the form of "srsRAN Project", not to be confused with "srsRAN" which is the 4G deployment of the RAN, which is an open-source implementation of 5G CU/DU from the company SRS. It is a complete RAN solution fully compliant with both 3GPP and O-RAN ALLIANCE specifications. There are multiple deployment scenarios for this project, but our end goal is a split 8 deployment. The aim was to implement a CU-DU O-RAN split. However, at the beginning, the the srsRAN Project install needs to be vetted off and one must ensure that communication with the Open5GS core and with the broader internet was smooth. To this end, a simpler deployment was proposed first, one that is not fully O-RAN compliant, having a monolithic gNB.

All components being installed and ready to use it is now time to configure communication between them. The first need would be to customize a range of files that are used to configure both the Open5GS core and the srsRAN Project-powered RAN.The first file to modify is the "amf.yaml" file that specifies the behavior of the AMF component of the Open5GS core. The file can be found in the folder "/etc/open5gs" alongside all other configuration files. Below one can find an example of such a configuration

```
# The IP address that the AMF component will use
to listen for connections from the srsRAN
  ngap:
    server:
      - address: 127.0.1.100[…]
# AMF id components
  guami:
    - plmn_id:
        mcc: 001
        mnc: 01 […}
# Tracking Area Identity
  tai:
    - plmn_id:
        mcc: 001
        mnc: 01
      tac: 7
  plmn_support:
    - plmn_id:
        mcc: 001
        mnc: 01
```

Furthermore, on the core network side of things one needs to configure the UPF component "upf.yaml" by setting up the GTPU server address.

For testing, the srsRAN has provided the gnb configuration at "gnb_rf_b200_tdd_n78_20mhz.yml". This can be used without any modifications, and if all steps were followed correctly it leads to a successful deployment of a monolithic gNB 5G Private Network.

Since release version 24.10, the srsRAN Project supports the CU/DU split 8 and implements two new applications for that. The "srscu", which can be found in the "srsRAN_Project/build/apps/cu" folder, and the "srsdu", which can be found in the "srsRAN_Project/build/apps/du" folder. Just like before, each app requires a YAML configuration file, which, in contrast to the previous deployment, where the example file had all the default values already dialed in, now the files need to be written from scratch. To help in this endeavor, the documentation provides a configuration reference [7] which describes all the possible fields that can be added and comments regarding their use alongside default values. It must be noted that the reference is not always updated at the same time as the project, and that there were cases in which we had to open a communication channel with the developers using a GitHub Issue with the creators of the library to get them to update the reference and provide guidance on the latest changes.

As there are two applications to run now, instead of a monolithic gNB, there will be two configuration YAML files. The first one will be used to describe the CU component, which has two individual parts, the CU-CP and the CU-UP, and a second one will be used to describe the DU component. Figure 2 represents the current setup and shows that the deployment closes in to being a fully O-RAN compliant 5G network deployment, the missing component at this point is represented by the RIC which will act as the housing component of the xApps that provide control over each of the RAN elements.

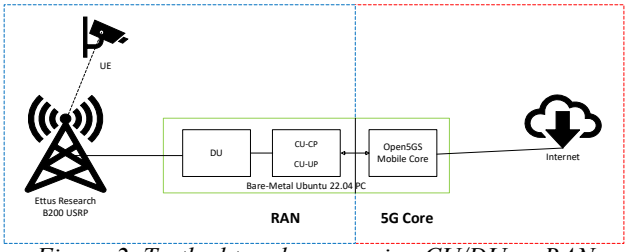

Figure 2. Testbed topology running CU/DU srsRAN Project setup with UE

There are multiple RIC frameworks out there that implement the E2 interface and that might be compatible with the RAN implementation of the srsRAN Project. However, two stand out as being recommended by the library itself in its documentation. These two are the O-RAN SC RIC and the FlexRIC projects. There might be some compatibility issues with other projects, such as the E2 interface by the srsRAN Project which is based on the R003-v03.00 version of the O-RAN specifications, thus only supporting the E2SM_KPM and E2SM_RC service models with a few limitations.

For the R2SM_RC service model the ”Control Service Style 2” is the only one supported. Whilst for the E2SM_KPM service model the monitoring period was limited to one second. Even though there are only six O-RAN metrics exposed, all five service styles are supported. Out of them, the following are of interest for this paper: “DRB.UEThpDl”, measuring DL throughput and “DRB.UEThpUl”, measuring UL throughput.

Any of the two frameworks proposed by the srsRAN Project were compatible with this implementation of the E2 interface, but due to better documentation on writing and deploying xApps it was chosen to move further with the O-RAN SC RIC framework. Typically, the installation of this framework was rather complex, and it required knowledge of helm charts and Kubernetes. There was, however, a minimal installation called “i-release” that was deployed as a multiple container application using Docker Compose.

The custom xApp in this deployment was based on the “simple_xApp.py” application. This was an example of an xApp that was able both to listen to information from the RAN components and to provide configuration commands to the CU/ DU components. On top of this, custom logic was added to dynamically allocate or deallocate PRBs to a user (see Figure 3).

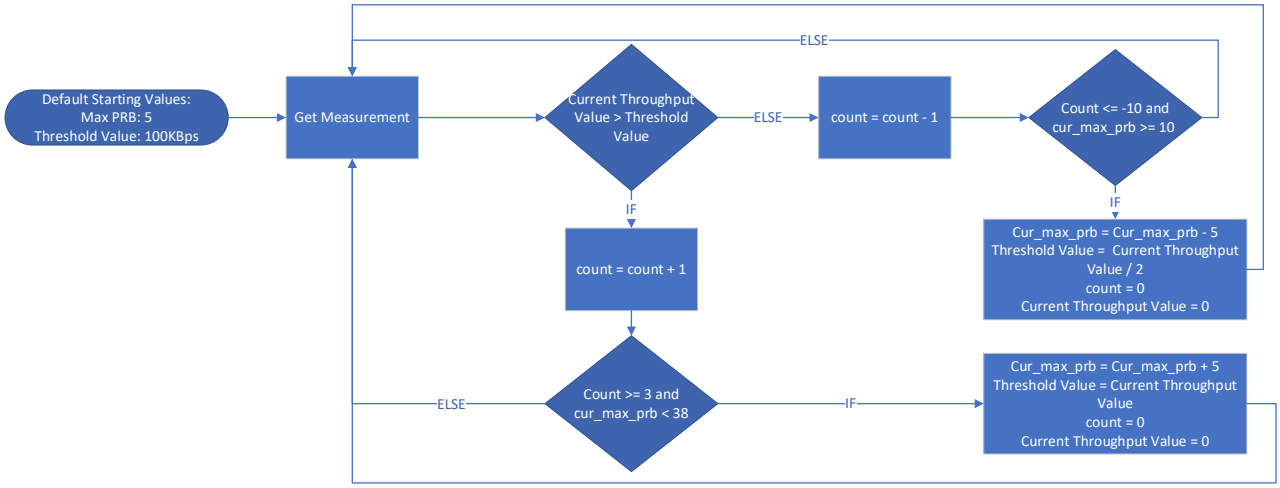

Figure 3. Custom logic function sequence diagram

The custom logic was rather simplistic and was mostly used to emphasize the fact that with open-source software and commercially available hardware one can build a real network. This could take decisions and change RAN parameters automatically based on real traffic data. The logic considered the amount of traffic through the RAN every second. If this value was constantly larger than the current threshold value it increased the number of PRBs allocated to that specific user. The reverse was true also, if the current throughput was constantly smaller than the threshold.

The value with which the number of PRBs was increased or decreased and the default starting values were chosen after multiple tests. We considered the most appropriate values to perform this experiment that generated a visible result. As the aim of this paper was to showcase a real end-to-end deployment of an O-RAN compliant private 5G network, there was less emphasis put onto designing the xAPP logic for performance, but rather onto placing such a system into a real environment.

The final topology is described in Figure 4. The 5G network and the customized xApp were now ready for the end-to-end testing.

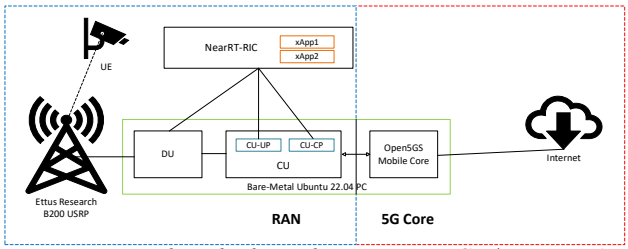

Figure 4. Final testbed topology running CU/DU srsRAN Project setup with RIC and UE

IV. EXPERIMENTAL RESULTS

For this first experiment the Nokia 5G Receiver was used with the APN set to bridge mode to test the maximum throughput at different distances from the USRP, using the speedtest.net website. The first thing to check was that the Nokia 5G Receiver connected to the 5G network had its signal level at an appropriate level. This was done in the GUI of the device which, if the settings were not changed, was accessed at 192.168.1.1, with the included username and password.

The Nokia 5G Receiver was placed at approximately 3m from the Ettus Research B200 USRP. The readout from the GUI showed an RSSI value of -74dBm and a SNR of 26dB (see Figure 5).

At this distance and with the current state of the channel and considering the programmed radio parameters (which allowed for a maximum of 38 PRBs to be allocated to a UE), the download speed had a peak at 32.86 Mbps and the upload speed at 2.83 Mbps. Decreasing the distance between the devices led to better connection, i.e., better RSSI (-66 dBm) and SNR (26 dB) values (see Figure 6) and an increase in the maximum throughput to peaks of 55.91Mbps on the downlink.

This stands to show that in the case of the devices used, even minimal factors like a distance difference of two meters allowed for almost doubling of the maximum throughput (attainable by a 5G deployment only in perfect conditions with fine-tuned devices).

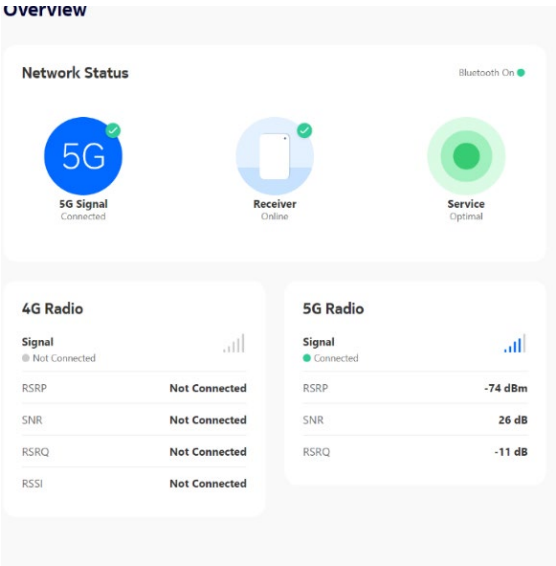

Figure 5. GUI of Nokia 5G receiver for Nokia 5G Receiver and B200 USRP@3m

This was not the case in this paper. Thus, it can also be expected that in the following experiments a degree of inconsistency might arise regarding the maximum throughput attained.

For the second experiment, the Nokia UE was replaced with a Milesight 5G CCTV camera, which was compatible with the 5G SA standard. The experiment was set in such a way that the distance between the camera and the B200 USRP was not greater than one meter for the best possible scenario.

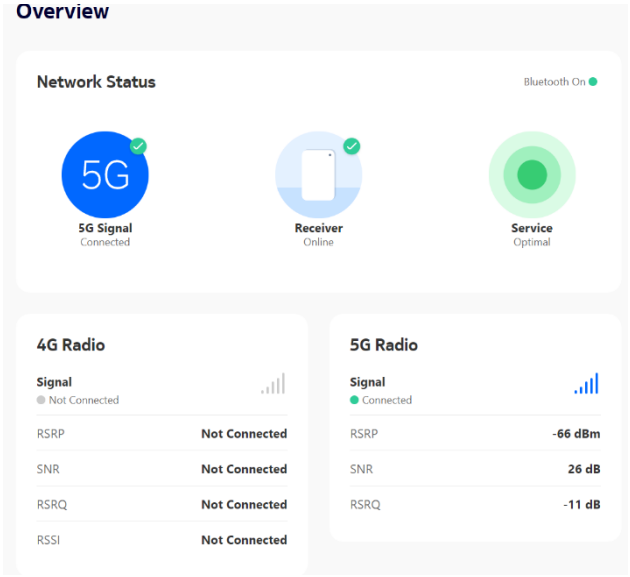

Figure 6. GUI of Nokia 5G receiver for Nokia 5G Receiver and B200 USRP@1m

The camera was pointing to a laptop which was running an action movie to make sure that the data coming in front of the camera was varied enough to generate higher amounts of traffic (see Figure 7).

Figure 7. Testbed setup for Dynamic PRB allocation with CCTV camera and B200 USRP@ 1m

For this experiment, the camera was set to transmit data using the H.264 encoding standard at a resolution of 1920x1080 (Full HD), with a maximum framerate of 25fps. The bit rate was set at 8192 kbps, which translated to a maximum of 1024 kBps, and the bit rate control was set to CBR mode. In this mode, the camera tried to attain the set bit rate, and even if that meant sacrificing the quality of the image in this case lowering the framerate. The PC that also runs the core network and the srsRAN project is then used to access the interface of the camera and this way traffic is generated over the radio interface and is picked up by the xApp that is running which starts to allocate more PRBs if necessary.

The results of the experiment were summarized in a couple of graphs that were generated using Excel by parsing the text output of the xApp using a simple Python

script. The data is parsed and placed in a CSV (Comma Separated Values) document.

The two resulting graphs show the evolution in time of the maximum number of PRBs that were allowed to be allocated to a particular UE in Figure 8.

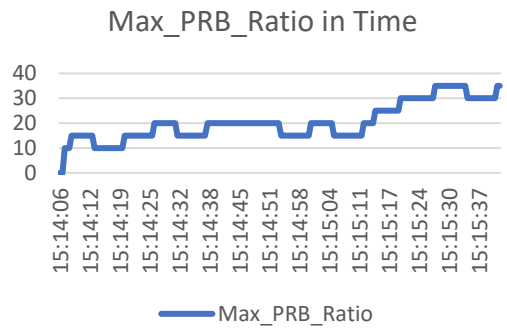

Figure 8. Maximum number of PRBs in Time CBR

The evolution in time of the throughput of the stream and the thresholds at which the switch happens, either up or down depending on the traffic to threshold mapping, is presented in Figure 9.

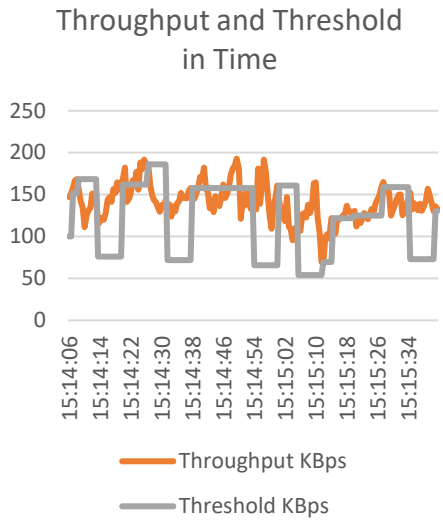

Figure 9. Throughput and threshold in Time CBR

From these two graphs, it can be observed that at about "15:14:22" the dynamic allocation mechanism caught up to the current need of the system, and the CBR mechanism kept a somewhat steady value of the bitrate for a few seconds. It can be sees that the dynamic allocation mechanism was trying to keep up with the "waves" in by the flow of traffic. Towards the end of the recording the mechanism started going up as a response to a sudden fall in traffic that led to the need for a quick recovery of the system, then it went back to a steady flow of traffic.

The third and last experiment used the same testbed setup as the second, but a change was made to the video streaming settings of the MileSight camera (see Figure 10). The bit rate control field was set to VBR mode, and the new image quality field was set too high to force the camera to push more traffic to the radio interface and to keep the bitrate higher. The results were interpreted in the same way, as can be seen in the two graphs generated (Figure 10 and Figure 11), there were more frequent switches in the maximum number of allocated PRBs as the bit rate varies considerably from one second to another.

Moreover, there are much higher spikes in bitrate, leading to steeper growth, especially at the beginning when the dynamic PRB allocation mechanism needed to catch up with the camera's needs.

Considering the inherent dynamic nature of the VBR mechanism whose goal was to push as much data as it could through the system, we had to take into consideration that it collided with our own mechanism for managing throughput. Thus, it quickly overpowered and saturated the connection, mostly due to the high frequency of the spikes in traffic.

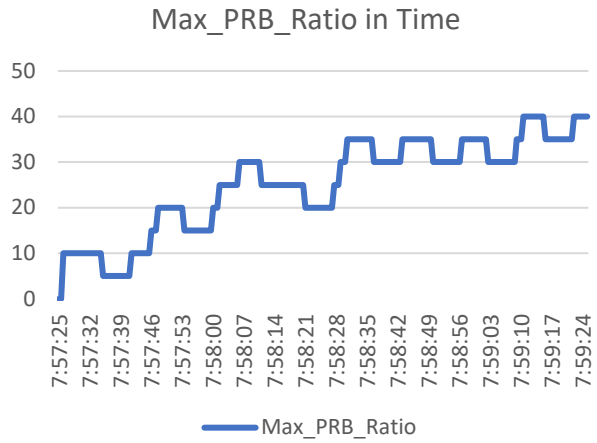

Figure 10. Maximum number of PRBs in Time VBR

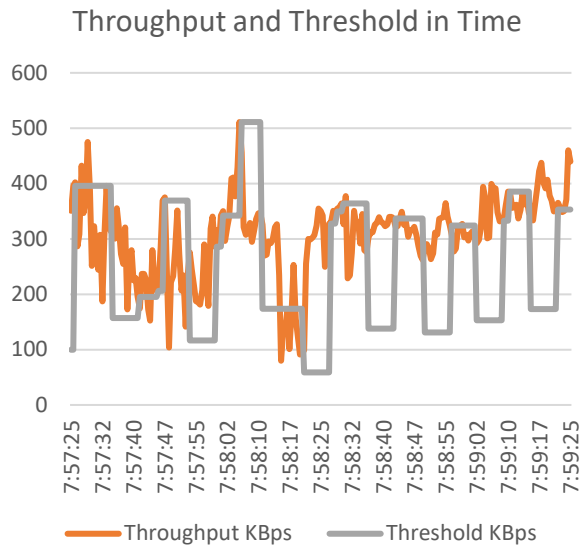

Figure 11. Throughput and threshold in Time VBR

Overall, in both experiments the dynamic PRB allocation mechanism did work but due to the empirical way in which the threshold setting mechanism was designed the system to overpower the VBR mechanism and to give up. Thus, it became redundant once the constant need for data kept the demand high. This was very useful

in times when the need for data was not as great, and it could "save up space" for other devices. As soon as more traffic hit the VBR mechanism and it started fluctuating heavily, it might break down.

In comparison when the system was dealing with the CBR mechanism the way of working was straight-forward. The main downside was the delay between the allocation of more PRBs and the decision of the CBR mechanism to try to switch to another level in bit rate which led to a longer response time between a need and its satisfaction.

V. CONCLUSIONS AND FUTURE WORK

This paper aimed to present a complete end-to-end deployment of a 5G Standalone Private network compatible with O-RAN ALLIANCE specifications. Also, a custom-built xApp demonstrates the advantages of using an O-RAN standard-based system even in real non-virtual deployments. It has based itself on the support offered by the srsRAN Project to be able to showcase a 5G RAN implementation that aligns with the OpenRAN concept, on a supported implementation of the 5G mobile core, i.e., Open5GS and on devices from various vendors that were interconnected by various pieces of software either by emulation or implementing the 3GPP defined standard interfaces.

The implementation hit various roadblocks in the format of lack of documentation, both regarding the software components like the srsRAN Project or the O-RAN SC RIC, but also in the hardware part. The Nokia 5G receiver advanced documentation is not freely available online and the Milesight CCTV camera lacks features that are specific to understand the current channel parameters in depth.

In future developments some flaws will be addressed using Artificial Intelligence/ Machine Learning algorithms. They can collect data as the system functions and retrain themselves using the deployment of a non-RT RIC.

Going forward these algorithms can be used to dynamically determine the PRB increase/decrease step or even the default starting values. Moreover, if traffic classification can be achieved in the 5G Core this information can be passed to the non-RT RIC and modify these algorithms to apply differentiated rules to specific devices or traffic patterns. In terms of hardware, with the move of the Core and RAN software to a newer platform and with improved RF hardware to allow the UEs to take advantage of MIMO technology we can increase throughput and get to speeds that rival operators led implementations that will allow for real-world testing. In turn, this opens the door for new xApp use cases and new research opportunities.

ACKNOWLEDGMENT

This paper was partially supported by the Project "Transforming the university ecosystem through digital transition towards a sustainable European future - eUT4ALL", Project code – e-PNRR: 1277457265. We acknowledge also the initial technical support from Orange Romania.

REFERENCES

[1] "Governance of O-RAN ALLIANCE e.V. in Compliance with WTO Principles, July 2023", O-RAN ALLIANCE, 2023, [Online], Available: https://mediastorage.o-ran.org/white-papers/O-RAN.O-RAN-ALLIANCE-in-Compliance-with-WTO-Principles-white-paper-2023-07-v02.pdf.

[2] S. Marinova and A. Leon-Garcia, "Intelligent O-RAN Beyond 5G: Architecture, Use Cases, Challenges, and Opportunities," IEEE Access, vol. 12, pp. 27088–27114, 2024, doi: 10.1109/ACCESS.2024.3367289.

[3] M. Hoffmann et al., "Open RAN xApps Design and Evaluation: Lessons Learnt and Identified Challenges," IEEE J. Select. Areas Commun., vol. 42, no. 2, pp. 473–486, Feb. 2024, doi: 10.1109/JSAC.2023.3336190.

[4] M.M.H. Qazzaz, Ł. Kułacz, A. Kliks, S.A. Zaidi, M. Dryjanski, and D. McLernon, "Machine Learning-based xApp for Dynamic Resource Allocation in O-RAN Networks," in 2024 IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN), Stockholm, Sweden: IEEE, May 2024, pp. 492–497, doi: 10.1109/ICMLCN59089.2024.10625184.

[5] "FastMile 5G Receiver 5G26-A", Nokia, 2025, [Online], Available: https://www.nokia.com/fixed-networks/fastmile-fwa/5g-receiver-5g26-a/.

[6] "5G AI 4X/12X Pro Bullet Plus Network Camera", MileSight, 2025, [Online], Available: https://resource.milesight.com/milesight/security/document/datasheet/ipc/milesight-5g-ai-4xand12x-pro-bullet-plus-network-camera-datasheet-en.pdf.

[7] "Configuration Reference", srsRAN Project, 2025, [Online], Available: https://docs.srsran.com/projects/project/en/latest/user_manuals/source/config_ref.html.

[8] G.M. Oltean, "OpenRAN Compliant Deployment in 5G Standalone with Customized xApp", Dissertation Thesis, Technical University of Cluj-Napoca, July 2025.