OPlaceRAN - a Placement Orchestrator for Virtualized Next-Generation of Radio Access Network

Fernando Zanferrari Morais, Gustavo Zanatta Bruno, Julio Renner, Gabriel Almeida,
Luis M. Contreras, Rodrigo da Rosa Righi, Kleber Vieira Cardoso, and Cristiano Bonato Both

Abstract—The fifth-generation mobile evolution enables transformations on Next-Generation Radio Access Networks (NG-RAN). The RAN protocol stack is split into eight disaggregated options combined in three network units, i.e., Central, Distributed, and Radio. Besides that, further advances allow the RAN functions to be virtualized on top of general-purpose hardware, using the concept of virtualized RAN (vRAN). The Combination of NG-RAN and vRAN results in vNG-RAN, which enables the management of the disaggregated units and protocols as a set of radio functions. However, the orchestration-based placement of these radio functions is a challenging issue since the best decision can be determined by multiple constraints involving RAN disaggregation, crosshaul networks requirements, availability of computational resources, etc. This article proposes OPlaceRAN, a vNG-RAN deployment orchestrator framed within the NFV reference architecture and aligned with the Open RAN initiative. OPlaceRAN supports the dynamic placement of radio functions focusing on vNG-RAN planning and is designed to be agnostic to the placement optimization model. To validate OPlaceRAN, we developed a prototype based on up-to-date cloud-native tools to deploy RAN using containerized virtualization using the OpenAirInterface emulator and considering two distinct functional splits (options 2 and 6). The evaluation is tested as proofs-of-concept in a real computing infrastructure using two different placement solutions. Our results reveal that OPlaceRAN is an effective cloud-native solution for containerized network functions placement and agnostic to the optimization model. Additionally, OPlaceRAN is up-to-date with the most advanced vNG-RAN design and development approaches, contributing to the evolution of the fifth-generation of mobile networks.

Index Terms—NG-RAN, RAN disaggregation, NFV, CNF placement, crosshaul networks

I. INTRODUCTION

The evolution of the fifth-generation of mobile networks is guided by transforming technologies and architectures into centric-software concepts to meet the new service demands, e.g., ultra-low latency and high data bit rate applications. For mobile operators, the softwarization process brings significant advantages, especially concerning time-to-market, usage of

Fernando Zanferrari Morais, Gustavo Zanatta Bruno, Julio Renner, Rodrigo da Rosa Righi, and Cristiano Bonato Both are with the University of Vale do Rio dos Sinos (UNISINOS). E-mail: {fmorais, zanattabruno, juliorenner}@edu.unisinos.br, {rrrighi, cbboth}@unisinos.br

Gabriel Matheus and Kleber Vieira Cardoso are with the Federal University of Goiás (UFG). E-mail: {gabrielmatheus,kleber}@inf.ufg.br

Luis M. Contreras is with the Transport & IP Networks - Systems and Network Global Direction, Telefónica CTIO Unit. E-mail: luis-miguel.contrerasmurillo@telefonica.com

commercial off-the-shelf (COTS) hardware, total cost of ownership (TCO) savings, adoption of open-source initiatives, continuous development and operations (DevOps), and network automation [1], [2]. Based on software concepts, the Radio Access Network (RAN) is in transition towards a general adoption of virtualization (i.e., vRAN) from the current situation in which RAN infrastructures are basically monolithic and highly distributed [3]. The most promising movement is led by the industry's Open RAN (O-RAN)[1] initiative, focusing on open and interoperable solutions [4]. Further advances are aligned with the proposed transformations towards the Next-Generation RAN (NG-RAN) architecture which allows the functional split of the radio protocol stack into eight potential options. These options can be combined into up to three network elements: *(i)* Central Unit (CU), *(ii)* Distributed Unit (DU), and *(iii)* Radio Unit (RU) [5]–[7]. These split options are intended to enable radio features and improve cost efficiency in comparison with previous generations of mobile networks [6], [8].

The functional splitting of NG-RAN combined with the virtualization in vRAN enables the vNG-RAN architecture to provide flexibility at the time of deploying mobile access networks [9], [10]. This flexibility allows the dynamic placement of radio functions following a fine-grained network management approach [11]. However, vNG-RAN introduces an unprecedented challenging problem since crosshaul transport networks (composed of backhaul, midhaul, and fronthaul networks) and computational resources (hosting the virtualized radio functions) have resource limitations, different resource dimensioning, and non-uniform resources consumption patterns [5]. In this context, the Network Function Virtualization (NFV) architecture leads the development and standardization of Virtualized Network Functions (VNF) in mobile networks to cope with the deployment and dynamic placement of vNG-RAN. Moreover, Container Network Functions (CNF) can be applied for radio functions based on the cloud-native approach and vRAN characteristics [4].

The NFV Orchestrator (NFVO) is an essential piece in the NFV reference model for placement and orchestration of the radio function chaining. NFVO is responsible for two main controls: Resource Orchestration (RO) and Network Service Orchestration (NSO), to support the VNF placement decision-making [12], [13]. Thus, the RO control concerns are related to

---

[1]https://www.o-ran.org/

the capacity and resources of the NFV Infrastructure (NFVI). Besides a relevant influence of the crosshaul networks (mainly latency and bit rate) capacity. Furthermore, the NSO manages the lifecycle of virtualized functions, particularly the chaining between them [14]. In this context, the O-RAN initiative is inspired by the NFV architecture, including the proposed Service Management and Orchestration (SMO) framework to guide the decision-making of the radio VNF placement [4].

The state-of-the-art related to placement and orchestration for vNG-RAN is diverse. On the one hand, placement's concepts are widely evaluated and massively developed in exact and heuristic approaches [15]–[19]. On the other hand, orchestration solutions are analyzed for generic functions not specific to the RAN environment. In this context, the article of Matoussi et al. [20] is an exception in the literature. However, the authors investigated the predecessor architecture of NG-RAN (i.e., C-RAN) using the OpenAirInterface (OAI) tool. In the industry, open-source orchestration platforms are being developed for generic VNFs [21], without specialization for vNG-RAN, to the best of our knowledge. The Open Network Automation Platform (ONAP) is an exception, working with O-RAN but without a concrete placement orchestration solution [22]. Moreover, it is fundamental to emphasize that a great tendency in the telecommunication industry standard applies the cloud-native approach [23] with the virtualization-based on containers, adopting Kubernetes (K8S) as the main container orchestration tool [24], [25].

Based on the diverse presented in the literature and industry open-source projects, this article proposes the Orchestrator Placement RAN (**OPlaceRAN**), a vNG-RAN deployment orchestrator framed within the NFV reference architecture and aligned to the O-RAN SMO framework. OPlaceRAN supports the agnostic placement of radio functions, focusing on the problem of vNG-RAN planning. Moreover, OPlaceRAN is designed following the functional NFVO sub-blocks, considering the RO control named **RANPlacer**, a complementary optimization module named **RANOptimizer**, the NSO control called **RANDeployer** and, finally, a data repository referred as **RANCatalogs**. RANPlacer handles the whole orchestration process, including the processing of external input from the Network Operator (quantity of radio units), crosshaul topology capacity, NFVI resources, and the alternative placement solutions stored in the RANCatalogs. RANOptimizer works with both exact and heuristics agnostic placement solutions aware of the functional split requirements. In this case, the agnostic solution is a strategy of vNG-RAN placement applied on the OPlaceRAN developed independently of the orchestrator. RANDeployer applies the virtualized radio functions addressed by the placement approaches according to the RANPlacer inputs according to the RAN CNFs also stored in the RANCatalogs. All the configuration, initialization, and validation processes of the virtualized radio functions are performed and activated by the RANDeployer.

The results obtained in our evaluation show that OPlaceRAN is a powerful orchestrator to leverage the placement of virtualized functions of vNG-RAN in NFV architectures. Our contributions can be summarized as follows:

- The definition of a conceptual orchestrator aligned with the NFV architecture and consistent with the O-RAN SMO framework for agnostic placement solutions for vNG-RAN.
- The development of a real prototype aligned with the cloud-native approach to develop vNG-RAN's virtualized radio functions considering the three units of vNG-RAN, i.e., vCU, vDU, and vRU.
- The realization of a proof-of-concept with real computational NFVI to provide the virtualized function placement analysis for vNG-RAN networks.

The article is organized into the following sections. Initially, Section II introduces the concepts developed in the article. Section III presents the OPlaceRAN architecture, and Section IV describes the prototype used in the experiment. Moreover, Section V shows the developed prototype and the proposed evaluation environment. Finally, Section VI brings the related work and Section VII concludes the article.

II. VIRTUALIZED NG-RAN ORCHESTRATION

The decomposition of the radio software stack into non-monolithic components aggregated in elementary functions is vital to improving RAN flexibility and processing performance for the 5G and beyond networks. The standardization initiatives such as 3GPP [9] and IMT-2020/5G from ITU-T [5] specify the number and types of functional splits. Therefore, the NG-RAN architecture aims to disaggregate and distribute radio functionality into three units (CU, DU, and RU). Moreover, the NG-RAN architecture flexibility allows some other configurations with less than three units, named DU and RU integration (DU and RU), C-RAN (CU and DU), and D-RAN (CU, DU, and RU in a monolithic way) [5]. In a disaggregated NG-RAN, the crosshaul transport network provides communication paths among radio units composed of different segments such as backhaul (mobile core to CU), midhaul (CU to DU), and fronthaul (DU to RU). Each functional split depends on radio capacity (e.g., channel bandwidth, modulation, and others), imposing distinct requirements in terms of latency, and bit rate for the crosshaul network [26], [27]. Consequently, the crosshaul must guarantee the latency and bit rate required according to the variety of radio functional splits [11], [26].

The adoption of virtualization concepts is essential for RAN disaggregation. Each virtualized radio unit (vCU, vDU, and vRU) is a VNF, resulting in a vNG-RAN architecture. The main reference virtualization architecture for telecommunication is the NFV architecture specified by the European Telecommunications Standards Institute (ETSI) [12]. In the NFV architecture, the Management and Orchestration (MANO) part is responsible for managing and orchestrating VNFs, defining such process as the automation, management, and operation of virtualized functions running on top of an infrastructure supporting virtualization and multi-tenancy. NFVO block orchestrates the NFVI resources across multiple Virtualized Infrastructures Managers (VIM) and manages the lifecycle of VNF services by the Virtual Network Functions Manager (VNFM). To this end, NFVO works with the RO control to provide access to the NFVI resources in an abstract manner, i.e., regardless of any VIM. Moreover, NFVO per-

forms the NSO control to create end-to-end services, composing different VNFs, and managing the topology of network service instances [28]. Fig. 1 presents the NFV architecture and the positioning of radio VNFs in the architecture.

Fig. 1. NFV reference architecture.
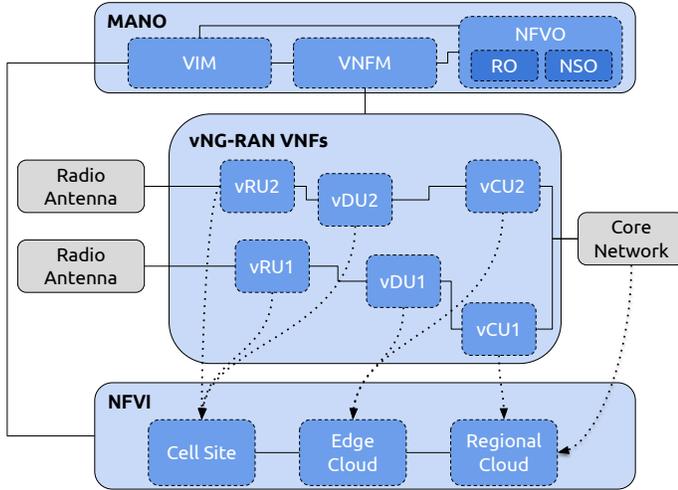
NFV architecture enables executing these radio VNFs over distributed Computational Resources (CR) interconnected through the crosshaul network. In this context, it is essential to highlight that O-RAN initiative also follows the NFV architecture. In this sense, O-RAN proposes the SMO framework focusing on slicing concepts, programmable RAN network resources, and embedded artificial intelligence capabilities [4], [10]. O-RAN uses both NFVO and VNFM MANO blocks to develop the orchestration and lifecycle of VNFs. In this sense, the SMO framework defines the RAN Intelligent Controller (RIC) to manage the RAN network divided into two components: (i) Non-Real Time (centralized components) and (ii) Near-Real Time (closed to vNG-RAN). Therefore, the SMO framework can connect to NFV architecture and provide a consolidated approach to the fifth-generation of mobile networks [4], [27].

To deploy RAN VNFs on top of NFVIs, the NFVO instantiate virtualized radio functions by leveraging hypervisor-based on virtual machines or containers. In this context, the container virtualization is targeted by O-RAN since it shares the Operating System that hosts the VNFs. Moreover, recent results show that the containerized solution tends to be the most appropriate for RAN virtualization due to the strict requirements of delay and scalability of resources. In this sense, each virtualized radio function is envisioned as a container [27], [29]–[31]. For the orchestration of containers in generic applications, four tools lead the implementations: (i) K8S, (ii) Docker Swarm, (iii) Mesosphere Marathon, and (iv) Cloudify. From the four options, K8S emerges in telecommunications virtualization as the main tool due to extensive Continuous Integration/Continuous Delivery (CI/CD) development, flexible configuration options (e.g., customized thresholds), and support for Docker-based containers (widespread). K8S focuses on managing virtualized infrastructures and orchestration in

the form of clusters managed over a pool of computational resources, whether physical or virtual [24]. According to that, the vNG-RAN placement orchestration solution is explored in the next section.

III. OPLACERAN: ORCHESTRATOR PLACEMENT RAN

This section introduces OPlaceRAN, i.e., an orchestrator with decision-making support for radio function placement. We design OPlaceRAN to overcome the lack of placement orchestrators for vNG-RAN. OPlaceRAN seeks to be an agnostic placement tool considering exact and heuristic placement solutions for the orchestration of planning of vNG-RAN. According to the NFVO functional block of the NFV architecture, OPlaceRAN operates between RO and NSO. In this context, OPlaceRAN is aware of crosshaul topologies. Moreover, the defined virtualization technology is based on containers, following O-RAN initiative to obtain the possibility of interoperability with tools that meet the evolution of the fifth-generation of mobile networks.

Fig. 2 shows OPlaceRAN positioned into NFV reference architecture for the RAN planning and design in the SMO framework. The overall NFV architecture components are covered, including VNFM, VIM, CNFs, and NFVI blocks in the end-to-end orchestration structure of vNG-RAN. In this sense, the OPlaceRAN design represents the intersection between the NFV architecture and the SMO framework, since SMO adopts NFVO and VNFM blocks from the NFV architecture. Moreover, it is worth to highlight that Near-Real Time is positioned outside the SMO framework, i.e., close to vNG-RAN (vCU/vO-CU, vDU/vODU, and vRU/vO-RU). Lastly, it connects to the Core Network and Radio Antennas (including UEs) and receives external input data from Network Operator and Crosshaul Topology.

OPlaceRAN is responsible for the entire orchestration of the system to place virtualized radio functions into NFVI, playing as an NFVO. The main goals of OPlaceRAN are: *(i)* receive input from Network Topology by Network Operator, *(ii)* analyze the available computational resources (CRs), *(iii)* support an optimization solution for planning RAN networks, and *(iv)* execute the deployment for positioning the virtualized radio functions. Moreover, OPlaceRAN interacts with the entire system. For example, OPlaceRAN communicates with VIM and VNFM blocks to collect infrastructure information and execute network scaling actions regarding MANO functions. In this context, OPlaceRAN adopts the indirect communication mode between VIM and VNFM blocks, as indicated in the NFV architecture [12]. For providing this indirect mode, OPlaceRAN is divided into three functional sub-blocks: RANPlacer, RANOptimizer, and RANDeployer. The three functional sub-blocks are described as following.

- **RANPlacer**: it is positioned as a RO functionality from the NFVO perspective, centralizing and determining the actions of OPlaceRAN for making decisions. To begin the vNG-RAN planning process, RANPlacer operates on demand from external inputs. RANPlacer receives external vRUs from Network Operator and Crosshaul Topology. Based on these external inputs, RANPlacer

Fig. 2. OPlaceRAN aligned with NFV architecture and set up with the O-RAN SMO framework.
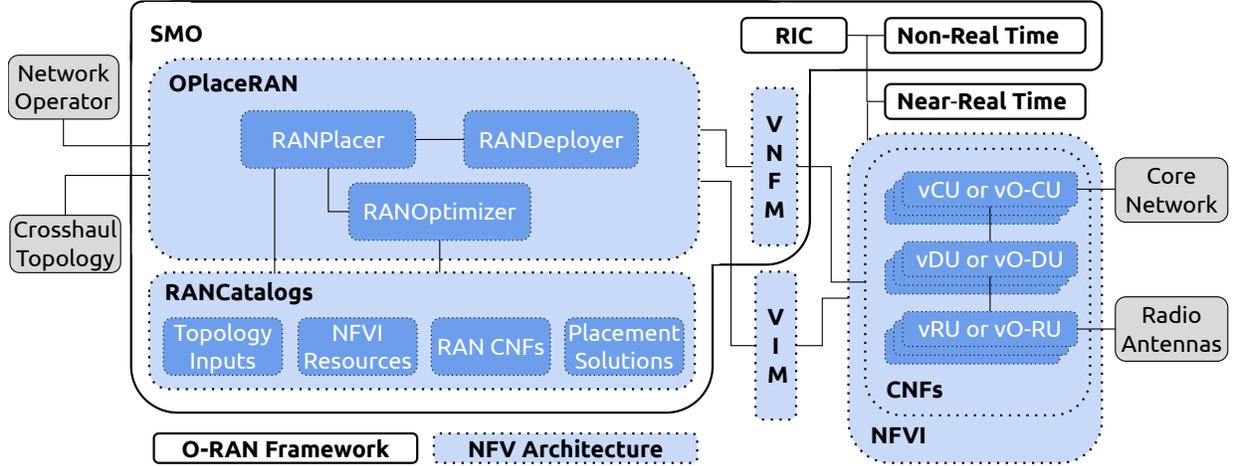
requests VIM block to update NFVI Resource Catalog. These inputs, together with NFVI resources, are sent to the RANOptimizer sub-block to analyze the optimal placement of the virtualized radio functions. The optimized result is analyzed by RANPlacer, which sends the VNF allocation and chaining plans to the RANDeployer sub-block.

- **RANOptimizer**: it is responsible for the decision-making process for the placement of virtualized radio functions. In this case, RANOptimizer receives requests from, and returns solutions to, RANPlacer. This request is made by inserting the RANCatalogs sub-block inputs and choosing a placement solution for runtime deployments. After that, RANOptimizer delivers the result of the radio VNF placement solution. In this case, RANPlacer receives from RANOptimizer a solution when feasible, including *(i)* position of vRU, vDU, and vCU units, *(ii)* disaggregation RAN options, and *(iii)* CRs to be allocated in the crosshaul networks.
- **RANDeployer**: it allocates the virtualized radio functions according to placement results from RANOptimizer sent to RANPlacer. In this case, RANDeployer operates as an NSO in NFVO, interacting with both VNFM and VIM blocks to deploy radio units. The interaction with the VNFM functional block assigns the creation and chaining activities of CNFs for virtualized radio functions. In addition, the interaction with VIM pursues the allocation of CNFs in the resources of CRs. Therefore, RANDeployer processes the requests from RANPlacer, requests radio units CNFs from the RANCatalogs sub-block, and sends the positions of CNFs to VNFM, deploying them in CRs of the NFVI.

To provide data support to OPlaceRAN operations, the **RANCatalogs** repository stores data about the overall services, functions, and resources available on the network. The RANCatalogs repository is composed of four different sub-catalogs. The first and second, i.e., Topology Inputs and NFVI Resources, operate as repositories of configurations. The Topology Inputs catalog works with external data entries

from the Crosshaul Topology inputs, e.g., network links and their respective latency and data rate capacities. Therefore, the Topology Inputs catalog is updated by the RANPlacer. The NFVI resources catalog refers to the available and allocated computing resources in NFVI. Examples of data types found in NFVI resources are CPU, memory, and storage. In this case, the update is based on the communication between RANPlacer and VIM. The third Catalog sub-block is Placement solutions. This sub-block supports a pool of the agnostic exact and heuristic approaches of placement solutions requested by RANOptimizer. Lastly, the RAN CNFs catalog operates as a repository of the container image registry for the radio units, i.e., vCU, vDU, and vRUs. The request for CNFs is triggered by the RANDeployer sending them to VNFM aiming to access the radio units and allocate them according to the guidance of the RANDeployer allocation.

Finally, Fig. 3 presents the sequence diagram of the OPlaceRAN workflow. The process starts with the collection of External Inputs event (step 01) of the Network Operator (NO) and Crosshaul Topology. Next, RANPlacer processes those inputs by considering the vRUs entry and the Crosshaul Topology (step 02). After that, RANPLacer sends a request to RANCatalog updating the VIM CRs (step 03), asking for an up-to-date view to NFVI (step 04 and 05) and receiving the update notification (steps 06 and 07). In the following interaction, OPlaceRAN requires an update for the Topology Inputs and NFVI Resources to the RANCatalog (steps 08 and 09). Therefore, RANPlacer sends the updated information and the external data to RANOptimizer, selecting the placement solution of RANCatalog and providing a feasible radio function placement (steps 10, 11, 12, and 13). After that, RANPlacer addresses the allocation plan to the RANDeployer function (step 14). Lastly, RANDeployer requires RANs CNFs Catalog images and sends them to VNFM for the CNFs chaining allocation (steps 15, 16, 17, and 18). Furthermore, the sequence diagram shows the interaction of the completion of the tasks between the NFVO module and the rest of components of the whole NFV architecture through the VIM, NFVI, and VNFM. The OPlaceRAN operation finishes with

Fig. 3. OPlaceRAN diagram sequence.

NO = Network Operator   CT = Crosshaul Topology
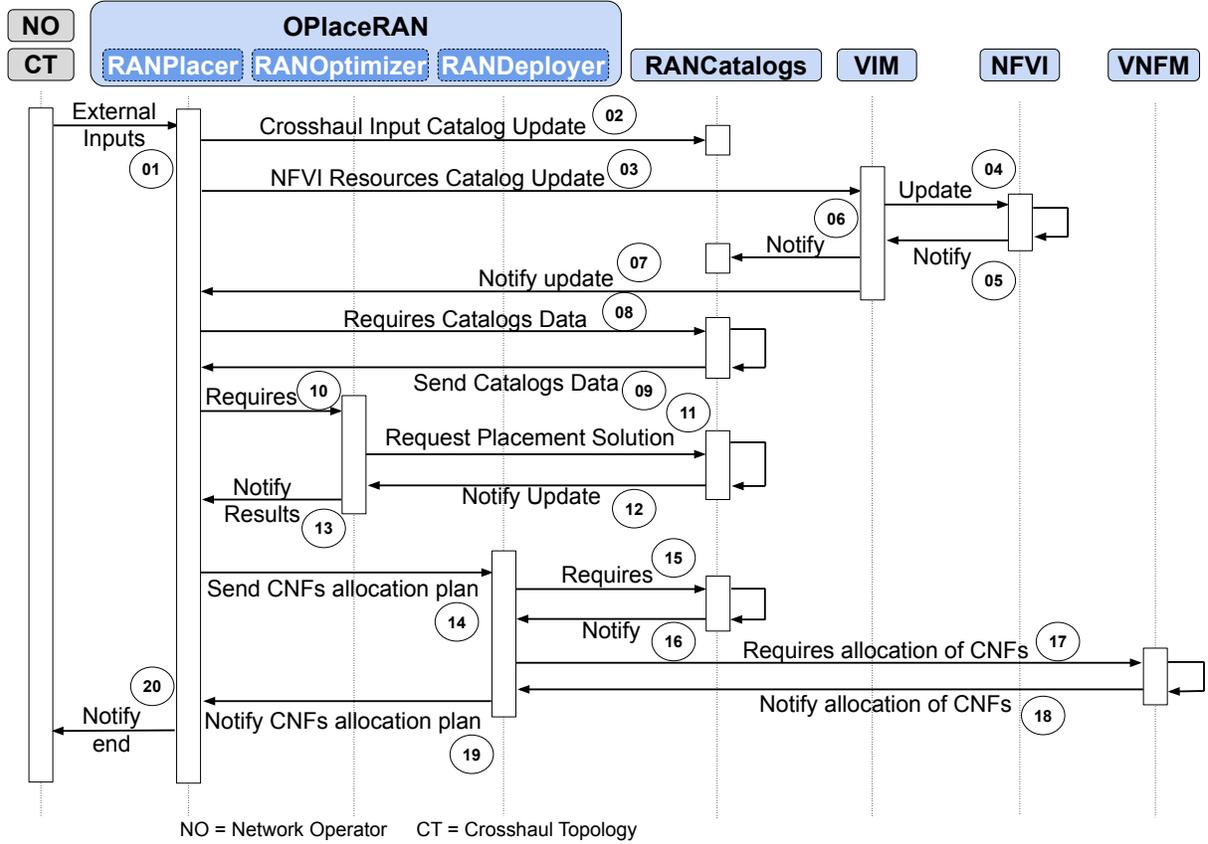
the allocation of the VNFs in CRs and its notification (steps 19 and 20). The following section describes a prototype to validate the OPlaceRAN approach on K8S, considering this sequence diagram discussed.

IV. OPLACERAN PROTOTYPE

To validate the OPlaceRAN proposition, we developed a K8S cloud-native tool prototype. Fig. 4 shows the prototype components which comprise the NFV and SMO blocks to support OPlaceRAN and their sub-blocks RANPlacer, RA-NOptimizer, RANDeployer, and RANCatalog data repository. The prototype covers all the aforementioned presented workflow, from the collection of external inputs (NO inputs and Crosshaul Topology) up to the CNFs chaining allocation process. In the prototype, RANCatalog maintains the Topology Inputs and the NFVI resources implemented with custom resources feature from Kubernetes and uses a Container Registry for the CNF's repository. The K8S tools include both NFVI and VIM natively and support VNFM and NFVO of the NFV architecture. Moreover, the K8S cluster comprises nodes concerning CRs and runs either directly on top of the hardware (bare-metal approach) or abstracted as a virtual machine (VM). In this context, applications are deployed and managed by K8S into Pod resources, holding one or more containers, i.e., CNFs. In the K8S architecture, Master Nodes provide the complete control of the cluster through a K8S control plane deployed in specific nodes. However, CNFs are running in

the Workload Nodes. The main components of this control plane are Controller Manager, Scheduler, Storage, and APIs. The NFVI cluster (K8S oriented) applied in the prototype is represented by VMs. The Calico plugin is used as a Container Network Interface (CNI) to provide an IP network connection between the Pods (i.e., radio functions). The entire prototype code is available at GitHub[2].

K8S is highly customizable, even considering its default components. For example, the control plane works with the Operators pattern to extend the Controller Manager and APIs. The operator pattern combines two extension points: the API server extension adding new resources to the K8S API and a controller monitoring and executing the defined resources. Therefore, an Operator can automate the whole lifecycle of the prototype, providing application-specific automation without changing the K8S default behavior. Moreover, Operators work under a Pod and are driven by the K8S object model. K8S uses additional layers of abstraction over the container interface to provide scale, resiliency, and lifecycle management capabilities. In this sense, both RANPlacer and RANDeployer orchestration sub-blocks are developed under the K8S operator's pattern in the Golang language. Operators manage the decision-making process of the solution. For example, RANPlacer builds the information base (i.e., the VIM data and placement result) for the RANDeployer to allocate CNFs

---

[2]https://github.com/my5G/OPlaceRAN

Fig. 4. OPlaceRAN prototype.
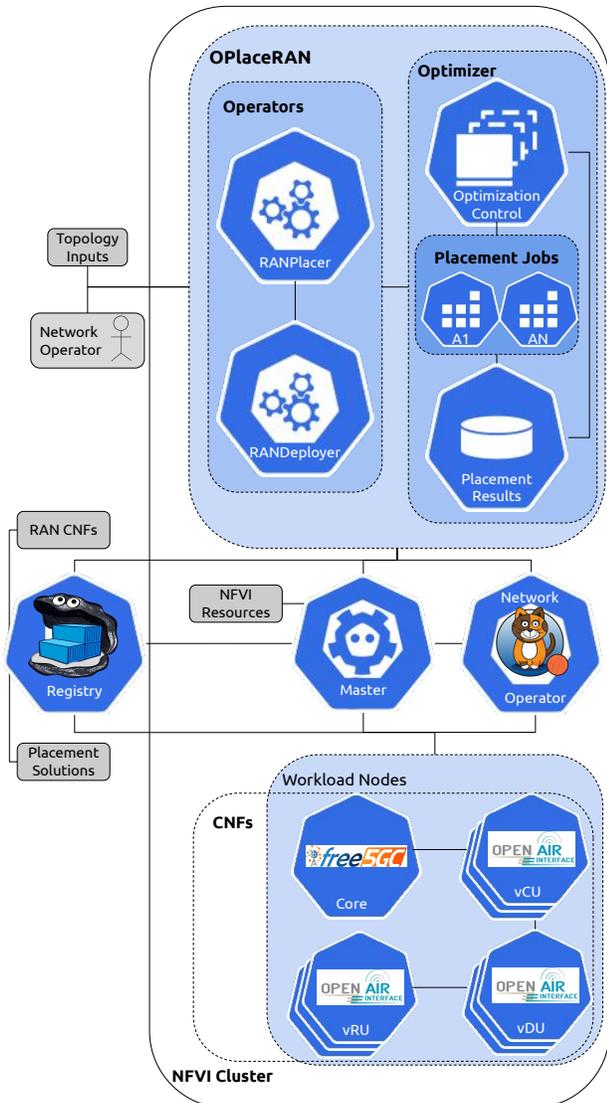
under Pods.

The Network Operator information includes the demand for vRUs and the Crosshaul Topology. The vRU demand is defined by the cluster of CR nodes, their vRU quantity, and the Crosshaul Input, one of three Catalog blocks defined as a ConfigMap. In the Catalog sub-block, NFVI Resources are acquired natively from K8S and matched with CRs of the nodes, i.e., the CPU and RAM capacity. The Placement Solution and the RAN CNFs Catalogs components are deployed in a Docker Registry server hosted in a virtual machine out of the NFVI cluster. The Docker Registry stores containerized images storing the placement solutions and the CNFs OAI images. These images are deployed on-demand by OPlaceRAN. We created images from available open-source projects: free5GC [32] for Core Network and OpenAirInterface (OAI) [33] for the RAN units. We only disabled the PHY layer to emulate RAN, considering all other layers to support the disaggregation on three units (vCU, vDU, and vRU), using the split options 2 (O2 - F1 interface) and 6 (O6 - nFAPI interface).

The RANOptimizer works based on three sub-blocks to deliver the placement results to the RANDeployer: (i) Optimization Control, (ii) Placement Jobs, and (iii) Placement Results. The Optimization Control receives requests from the RANPlacer with the inputs and the Placement Solutions. The input is composed of the following pieces of information: (i) CR nodes, (ii) number of chains that must be deployed, (iii) placement of vRUs in CRs, (iv) crosshaul topology, (v) capacity of the CRs (CPU and RAM), and (vi) specific placement solution (either the exact or heuristic approaches). The Optimization Control works similarly to an HTTP server. Therefore, the Optimization Control asynchronously triggers the placement solution job execution and provides to the RANPlacer a token used to get the placement solution status and the result. The Placement Jobs are also based on asynchronous tasks since they can take a considerable time to finish. In this case, a Job resource handles the execution of the placement solution, starting one or more Pods to provide a task-based workflow expected to run successfully after completing their goal. According to the selected placement solution by the Network Operator, a job is triggered by the chosen solution. The Placement Solutions are deployed as container images stored in the Registry repository. Once the job finishes its execution, it saves the placement results to the persistency layer accessible by the Optimization Control. Finally, the result of the placement solution is returned to the RANPlacer to be suitable for sending towards RANDeployer for the CNFs allocation process. In the next section, we evaluate the prototype in an experimental topology.

V. EVALUATION

This section evaluates OPlaceRAN in a Proof-of-Concept. Subsection V-A provides a general description of the evaluated scenario, detailing the placement solution applied, including topology, and resources, by varying parameters to assess the solution results. Subsection V-B presents the results obtained in the evaluation of OPlaceRAN concerning placement solution results, cluster, and nodes behavior. Lastly, Subsection V-C discuss general aspects and relevant characteristics of OPlaceRAN.

A. Scenario Description

We organized the assessment of OPlaceRAN into four parts, as described in the following. The first part presents the topology used, while the second shows the computational capacity of the resources. In the third part, the parameters used in vNG-RAN are explored. Finally, the fourth part details the placement solutions used.

- **Topology**: we run the experiments in a K8S cluster in version 1.19.3, with six nodes. Each node of this cluster is hosted in a VM running on a VMware ESXi 6.7 hypervisor. The six VMs run Ubuntu 18.04 with a low latency kernel, in which two nodes have the master control plane (no vNG-RAN was deployed at this stage) and the other four nodes have the worker role. VMs are hosted in three DELL PowerEdge M610 servers equipped with two Intel Xeon X5660 processors

running at 2.80 GHz and 192 GB of RAM. The Crosshaul Topology applied in the evaluation includes a physical switch (pSwitch) connected with the servers through a virtual switch (vSwitch) in the border. Each crosshaul link has a 10 Gbps reserved bandwidth and 1ms of latency guaranteed, with exception for the worker node six, which connects directly with the pSwitch and has 1 Gbps of bandwidth. Links latency between vSwitches and pSwitches are not considered. 5 shows the topology used, which supports four NG-RAN scenarios considering the units and nodes distribution, as addressed in the RAN Units item (described below).

Fig. 5. Experimental topology.
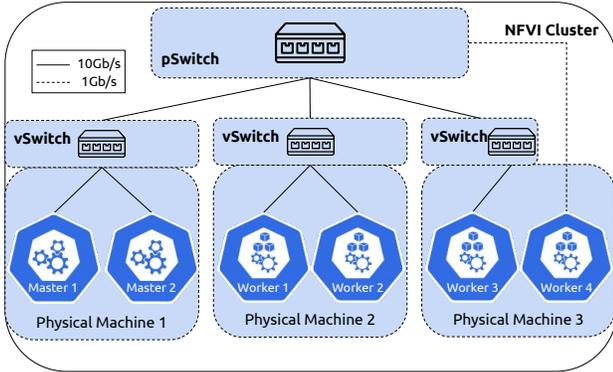
- **Computing Resources**: we focused on the processing capacity (i.e., CPU and RAM) because this metric is the most common bottleneck for computing devices in the context of vRAN evaluation, according to the OAI analysis [34], [35]. CR values in Table I indicate the number of available CPUs (Core unit) and memory RAM (GB unit) in worker K8S nodes at the time of generating input data. We fixed these values to have control under the deployment scenario, but our operator can automatically collect these values in the real world in placement time. The heterogeneity of CRs capacity used adds some complexity to the placement solutions that we evaluate. These capacities of CRs considered the CPU utilization profile of the RAN software as obtained from OAI implementation. The exact values may vary according to the adopted software components and the computing device, but similar profiles are employed in different works [34], [35].

TABLE I
RESOURCES TO RAN CNFs ON K8S WORK NODES.

Node	1	2	3	4
CPU (Core number)	5	1	2	1
Memory(GB)	2	0.2	1.2	0.2

- **RAN Units**: We defined a vRU for each work node. The parameters used in the two deployed placement models follow the definitions of the 3GPP [9], and ITU-T standardizing organizations [5], and the OAI Long Term Evolution (LTE) tool support, according to Table II. We use four NG-RAN scenarios regarding the units

and nodes distribution, i.e., *(i)* three units divided into vCU, vDU, and vRU; *(ii)* two units co-located vCU and vDU, *(iii)* C-RAN scenario, using two units and integration of vDU and vRU; and *(iv)* D-RAN with all the units in a single node. Additional parameters shown in Table II are latency and bandwidth supported by the OAI emulator for RAN functional split. In this sense, the Core Network (CN) parameters represent the requirements of communication between CN and vCU. In this work, we apply two RAN split options, namely O2 and O6 [36]. In Table II, the splits are represented by parameters O2 and O6. The O2 split parameters represent the vCU-vDU communication, and the O6 split parameters reflect the vDU-vRU communication.

TABLE II
LTE PARAMETERS OF RAN FUNCTIONAL SPLIT [36].

NG-RAN Scenarios	Latency - one way (ms)			Bandwidth (Mbps)		
	CN	O2	O6	CN	O2	O6
NG-RAN 3 units	30	30	2	151	151	152
NG-RAN 2 units	30	30	-	151	151	-
C-RAN	30	-	2	151	-	152
D-RAN	30	-	-	151	-	-

- **Placement Solutions**: we used two different placement solutions approaches to test the effectiveness of OPlaceRAN and the support for agnostic placement solutions. First PlaceRAN [37], an exact optimization approach focused on maximizing the aggregation level of RAN CNFs and minimizing the number of CRs used for such aggregation. The second is another exact optimization approach that combines a linearization technique with a cutting-planes method [15]. The focus is to apply the Multi-CUs vRAN concept to minimize the vRAN cost and overall routing based on CU's positioning. Both approaches explore the optimal solution of placement. It is fundamental to highlight that our goal is not to investigate the performance evaluation of placement solutions but the agnostic support for these solutions.

B. Results

We evaluate OPlaceRAN as proof-of-concept to obtain results that could indicate the scenario behavior using the orchestrator developed. Therefore, first, we detail the RAN CNFs allocation plan based on the placement solutions presented in the previous subsection. Second, we present the behavior of clusters and nodes. Finally, to evaluate the cluster's performance, we use just the PlaceRAN [37] placement solution because the focus is just the behavior of OPlaceRAN and not the performance of the placement solutions.

Regarding the level of aggregation of CNFs, which indicates the capacity of putting RAN unit nearest to the Core Network and in the same CR, respecting the requirements of crosshaul networks, mainly driven by latency and bit rate [37]. The results for the two placement solutions, PlaceRAN [37], and Multi-CUs vRAN [15] are presented in Fig. 6. The mathematical modeling of these placement solutions uses the same number of nodes, and each node operates with

only one RU (in the present work, only vRUs). PlaceRAN presents a slightly better performance because it aggregates more CNFs than the Multi-CUs vRAN model. Moreover, PlaceRAN concentrates on aggregating vCUs and vDUs CNFs only in Nodes 1 and 3, while Multi-CUs vRAN aggregates in Nodes 1, 3, and 4. In this analysis, PlaceRAN presents a better performance in the centralization of CNFs than Multi-CUs vRAN. This centralization is due to the flexible allocation of vDUs designed in PlaceRAN. The placement of DUs is predefined and inflexible in multi-CUs vRAN.

Fig. 6. Comparison of Multi-CU vRAN [15] vs PlaceRAN [37] solutions.
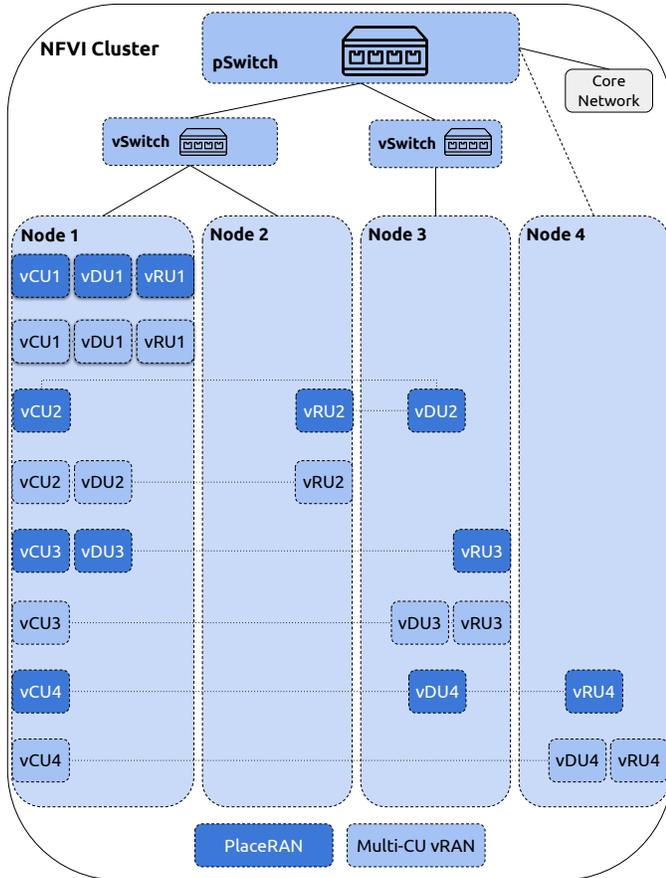
According to the placement solution results, the analysis of cluster computing behavior shows the relationship between the main events during the deployment of OPlaceRAN based on the use of CRs of the cluster. In this analysis, we only evaluate the PlaceRAN placement solution once the focus is on the behavior of OPlaceRAN and CRs. The events for a full deployment are highlighted in seven moments of time ($t0$ up to $t7$) as shown in Table III. Initially, the time consumed by the entire OPlaceRAN, from the collection of the external inputs (sent by Network Operator) up to the allocation of CNFs, is presented between $t0$ and $t1$s shown in Fig. 7. The time between $t0$ and $t1$ last approximately 70s being divided into six main events. The first event occurs with OPlaceRAN Start, beginning with the Network Operator and immediately RANPlacer Start to manage the process.

After that, the placement solution begins processing the inputs Network topology, vRUs locations, and node resources. The RANOptimizer Complete processing time takes approximately 1.2s after starting the placement due to the low complexity of the topology composed by three nodes. Following that, RANDeployer processes the placement solution result and sent for *RANDeployer Start*. These steps take around 31.5s. In the next step, RANDeployer sends the plan to the K8S with the configuration to be applied. Finally, K8S allocates the CNFs into the Pods (vCUs, vDUs, and vRUs) images to run on K8S nodes. When K8S Received CNFs allocation plan occurs, nodes download the OAI images and CNFs Pods are started (34s), the OAI is loaded by the container layer and, finally, all *CNFs become allocated* (70s).

TABLE III
KEY OPERATOR'S ACTIVITIES IN FUNCTION OF TIME.

Time	Description
$t0$	Started OPlaceRAN
$t1$	Ended OPlaceRAN, CNFs allocated
$t2$	Configured OAI network parameters
$t3$	OAI layer is loaded
$t4$	OAI functional and connected to Core Network
$t5$	Configured network tunnel between vRU and Core Network
$t6$	Ping from simulated UE to Core Network started
$t7$	Ping from simulated UE to Core Network terminated

Fig. 7. Initial OPlaceRAN steps.
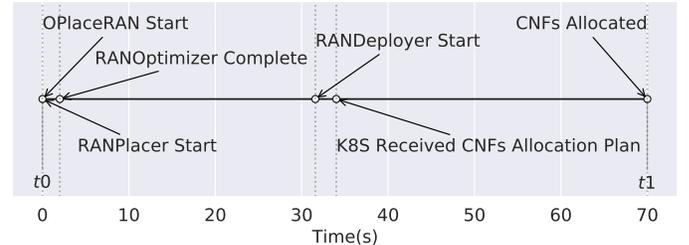
After the OPlaceRAN tasks are performed, it is necessary to configure the radio units through the parameters provided by the OAI tool for each RAN function (i.e., vCU, vDU, vRU) after all Pods are allocated and before starting the RAN CNFs. This configuration happens between $t1$ and $t2$ using the maximum CPU, near to 2000 CPU Millicores (i.e., two vCPUs), shown at the top of Fig. 8. After radio units are configured, the load of OAI layer is finished in $t3$. Next, OAI has already been configured and loaded, OAI processing is started. In this sense, the memory consumption in the K8S cluster increases significantly between $t3$ and $t4$, shown at the bottom of Fig. 8. Following the $t5$ time represents the establishment of the tunnel between vRU and the Core Network. In the $t6$ event, a network tunnel between the simulated UE and Core Network is established, and data traffic from the simulated UE to Core Network using the Ping application starts with default configurations for the 60s. We can observe a slight increase of CPU at the top of Fig. 8 between $t6$ and $t7$. Finally, it is fundamental to highlight that the CPU and Memory consumption of OPlaceRAN is insignificant. We can

see in Fig. 8 that the OPlaceRAN overhead hardly varies as a function of times and tasks running on the K8S cluster. In general, CPU and memory of vDU and vRU stabilize from $t5$ with the constant data packets generation, differently of vCU that did not vary after $t3$ because it only works as a traffic tunnel support for data packets after OAI is configured. One of the reasons for this behavior is that the OAI version works with emulated UE (without PHY protocol), and these UEs are fixed, reducing the network's signaling. This behavior was not considered as a trouble because of the paper focus.

Fig. 8. Cluster Computing Resources usage behavior.
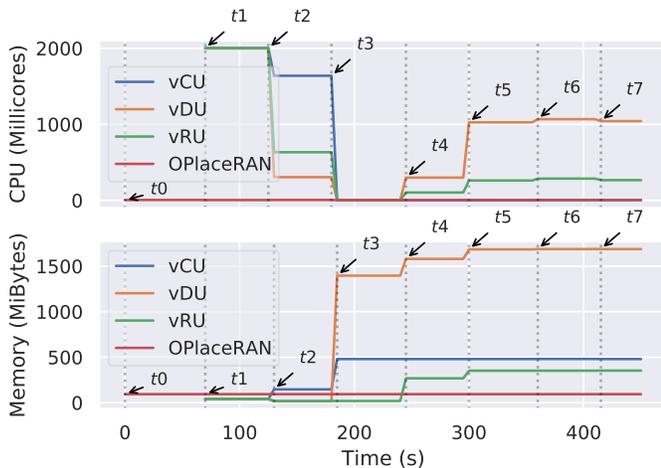
We must consider Table I to understand the CR consumption in each node of the K8S cluster. Firstly, we show the amount of CPU and memory that is guaranteed to be available to the CNFs placement, i.e., the input values to the PlaceRAN placement solution. The CRs diversity is necessary to produce placement solutions that have different behavior. In this case, the objective was to simulate an environment with scarce resources, with the placement solutions having RAN units behavior so diverse. Node 1 running in the network provided in Fig. 8 presents the highest average consumption of CRs during the experiment execution, 1053 CPU (Millicores) and 2135 MiBytes of Memory, as shown in Fig. 9. It makes sense because this node has higher CRs (5000 CPU Millicores and 2000 MiBytes of free memory), and all tested placement solutions prioritize to allocate the CNFs in Node 1. Both Nodes 2 and 4 allocated just one vRU because they have lower free CRs (1000 CPU Millicores and 200 MiBytes of Memory). Moreover, PlaceRAN allocated just one vRU because these CNFs use lower CRs than others, as seen in Fig. 7. Finally, Node 3 has an intermediate capacity of free CRs (2000 CPU Millicores and 1200 MiBytes of Memory). Furthermore, this node has a medium average consumption of CRs 613 CPU (Millicores) and 1448 MiBytes of Memory, but it has the highest allocation of proportional usage of CRs. This behavior is a consequence of the placement solution PlaceRAN because the Multi-CU vRAN allocates a lower number of CNFs in this Node 3. A relevant observation is the consumption of the Master nodes. Given the proof-of-concept scale, the four

vRANs chains present in the cluster did not generate a high workload for the K8S control nodes. Master 1 has higher CPU and Memory usage because it is the primary K8S control plane.

Fig. 9. Nodes resources average behavior.
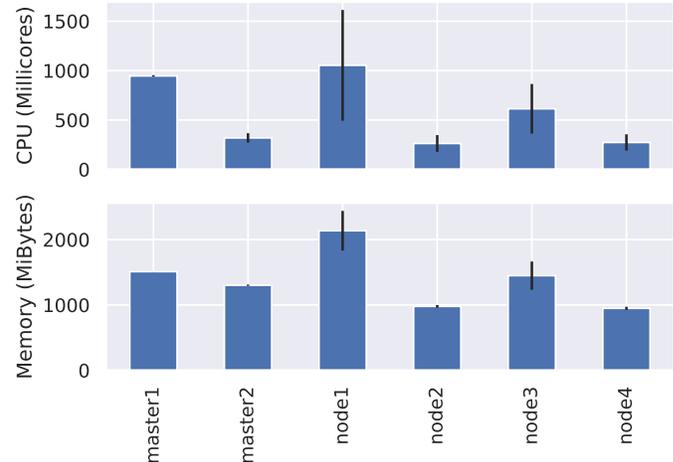
C. Discussion

Initially, we highlight OPlaceRAN within its conceptual design, including the prototype description, and present the development of vNG-RAN. Four pillars are crucial for the solution to achieve great potential: (*i*) alignment and positioning with the up-to-date architecture and frameworks of vNG-RAN, (*ii*) agnostic placement of CNFs approach, (*iii*) real prototype update with main tools and emulators, and crosshaul topology analysis. Furthermore, the first item opens opportunities for the integration of OPlaceRAN to any other solutions also aligned with the NFV architecture and SMO framework, whether this is within the scope of vNG-RAN itself or with the CNs' crosshaul transport networks.

Concerning the possibility of operating in an agnostic way for placement approaches, OPlaceRAN stands as a unique solution. Such an operation provides an opportunity to analyze different placement solutions, as performed in Subsection V-B, which compares two different strategies for the vNG-RAN placement. Furthermore, the possibility of exploring different placement solutions brings near-real operational network results. This approach differs from traditional mathematical placement solutions as they do not evaluate realistic NFVI infrastructures and containerized RAN applications. In this sense, i.e., the study of the NFVI cluster and its respective nodes, Subsection V-B makes it possible to add an assessment of the impact on CRs based on their hardware, operating system, functions, and consumption characteristics of native tools (e.g., K8S Master control and CNI Calico), and the deployment of the CNF allocation concurrently. Therefore, to bring a real impact to the performance of NFVI and RAN CNFs, in some cases, demands reassessments in the mathematical exact and heuristic approaches of placement solutions.

Finally, the interconnection of vNG-RAN through crosshaul networks, currently one of the biggest challenges of disaggregated RAN, can be performed by OPlaceRAN. Due to the size of the topology investigated in the proof-of-concept, the chaining of CNFs vCUs, vDUs, and vRUs present very similar latency values. Overall near 18ms on average for the three scenarios presented by the placement in the end-to-end communication approaches between UE and CN (analysis performed within the mobile network, not evaluated in external connections). Two relevant aspects of the latency's similarities are the number of hops (low processing latency) and the physical distance between nodes (all nodes in the same environment). Furthermore, since during the tests occurred various latency peaks, to understand this, we have to consider certain aspects: (*i*) we calibrated and left some slack the CRs available for the Core Network and RAN Pods, and (*ii*) we customized the OAI version as container-based with two splits disabling the PHY protocol. Finally, we have not investigated the bandwidth parameter in the evaluation because the maximum obtainable value is not relevant to generate an impact on the network (around 150 Mbps at maximum capacity, as shown in Table II). However, OPlaceRAN supports bandwidth configuration.

VI. RELATED WORK

Recently, RAN faced an intense process of *softwarization* and virtualization, leading to a broad transformation. In this scenario, the orchestration process has great relevance for the development of vNG-RAN. However, the research about virtualized RAN network functions orchestration has generated a divergent view between several works. Mainly concerning supporting the characteristics and requirements of vNG-RAN and the subsequent placement solutions strategies in the experimental orchestrators. In this context, the experimental orchestration studies are classified into investigations by open-project organizations and academia. For open-source projects, a strong alignment with the NFV architecture and the RAN disaggregation is observed. The projects Open Network Automation Platform (ONAP) [38], Open Source MANO (OSM) [39], Mosaic 5G [40] stand out in this scenario, led by the organizations Linux Foundation, European Telecommunications Standards Institute (ETSI), and EURECOM, respectively. For example, the project under development by ONAP is natively oriented by O-RAN initiative. Nevertheless, open-source projects still lack concepts aware of placement and crosshaul networks.

For academic purposes, we developed Table IV to guide the analysis of works, showing the state-of-the-art orchestration, placement solutions, NFV architecture, CNF support, and RAN disaggregation. The orchestration characteristic is the most relevant when comparing the related work since there are investigations with different goals. In some studies, orchestration focuses on the fronthaul performance [41]–[43], while other works prove concepts related to the framework's functioning without an orchestration strategy [22], [42], [44]. However, Dalla-Costa et al. [45] and Matoussi et al. [20] provide consistent orchestration goals. The first work introduces a load balancing algorithm with an analysis of the

performance of the fronthaul network and the computational resources. The authors use a synthetic C-RAN, with the behavior results closely regular and constant. The second work developed a particular heuristic approach for the placement of radio functions to reduce the consumption of computational resources. Based on the works presented, OPlaceRAN is the first work in the literature that supports agnostic placement solutions, enabling different placement strategies, empowering the planning of vNG-RAN. Moreover, OPlaceRAN deals in an integrated way with CRs strategies and topologies features for crosshaul networks.

Two studies about virtualization are aligned with the NFV architecture and support for container technology. The NFV architecture is an essential guide to the virtualization process because its reference model integrates several parts of fifth-generation of mobile networks, e.g., CN and RAN integration [2]. Moreover, container virtualization support is the leading solution for the granularity of virtualized radio functions [27]. In this sense, Rodriguez et al. [22] present a demonstration in line with the NFV architecture and meet container virtualization under the K8S tool. However, the authors only provide the RAN disaggregation for NG-RAN with three radio units guided by O-RAN concepts, without taking into account the crosshaul networks, and do not consider any placement solutions. Therefore, despite all the ongoing development on virtualized architectures for the fifth-generation mobile networks, OPlaceRAN fills an open research gap regarding an orchestrator with vNG-RAN placement support aligned with NFV architecture, directives of the O-RAN projects, and evolution from RAN to vNG-RAN. Furthermore, OPlaceRAN is aligned with up-to-date virtualization tools and technologies based on CNFs, including functional RAN splitting requirements, awareness of the crosshaul topology and computational resources, and providing opportunities for the integration with other platforms further than the mobile networks.

VII. FINAL REMARKS

In this work, we have proposed and validated the OPlaceRAN solution for orchestration of placement optimization solutions of virtualized disaggregated radio functions guided by the standards defined for vNG-RAN and aware of the requirements crosshaul networks and computational resources. Moreover, the OPlaceRAN concept has native support for generic and agnostic placement solutions approaches, providing opportunities for different solutions strategies to leverage the RAN planning evaluations. Furthermore, the concept aligned with the NFV architecture and set with the O-RAN framework brings an up-to-date position of the work in the RAN research. Finally, a prototype based on the cloud-native, CNFs concepts, and K8S tool are developed to validate the conceptual orchestrator.

OPlaceRAN was evaluated in an experimental scenario based on the deployment of two different placement solutions in the proof-of-concept. In this sense, we proved an effective and agnostic optimization model in a testbed experiment, emulating a mobile network with open-source tools. The conceptual solution and experimental results showed the effectiveness of OPlaceRAN in the orchestration of control

TABLE IV
RELATED WORK - ACADEMIC PLACEMENT ORCHESTRATORS.

Works	Orchestration Goals	Placement Solution	NFV Architecture	CNF Support	RAN Disaggregation
[46]	Native K8S tool orchestration	✗	✗	✓	Not specified
[42]	High availabilty fronthaul orchestration	✗	✓	✓	C-RAN
[43]	Fronthaul driven RAN	✗	✗	✗	C-RAN
[44]	Manual orchestration of VNF and PNF	✗	✓	✗	NG-RAN 2 units
[22]	Demo orchestration under ONAP platform	✗	✓	✓	NG-RAN 3 units
[41]	Wireless fronthaul performance driven orchestration	✗	✓	✗	NG-RAN 2 units
[45]	Resources load balancing orchestration for C-RAN	Specific	✓	✓	C-RAN
[20]	Specific placement orchestrator for C-RAN	Specific	✗	✓	C-RAN
OPlaceRAN	Agnostic placement orchestrator for vNG-RAN	Agnostic	✓	✓	NG-RAN 3 units

and deployment of placement solutions strategies for vNG-RAN. The entire orchestration steps, cluster/nodes resources, and crosshaul requirements are controlled by the OPlaceRAN orchestrator with a low computational overhead. Finally, we can conclude that OPlaceRAN contributes to the vNG-RAN transformation and opens opportunities to integration with other fifth-generation of mobile network platforms.

For future work, the OPlaceRAN opens several opportunities. First, we envision adding to the orchestrator an integration of flow forwarding controller by the Software-defined Networking (SDN) to leverage and automatize the crosshaul network's capacity and evaluate OPlaceRAN in a topology with more nodes and complexity. The other relevant goal is to connect the core network standalone standard and integrate it with RAN placement strategies. Moreover, there is the need for introducing the RAN slicing concepts aligned with O-RAN.

ACKNOWLEDGMENT

REFERENCES

[1] Y. Zhang, *Network Function Virtualization: Concepts and Applicability in 5G Networks*. John Wiley & Sons, 2018.

[2] GSMA, "Migration from Physical to Virtual Network Functions – Best Practices and Lessons Learned," 2018.

[3] M. Liyanage, A. Gurtov, and M. Ylianttila, *Software defined mobile networks (SDMN): beyond LTE network architecture*. John Wiley & Sons, 2015.

[4] O. R. Alliance, "Cloud architecture and deployment scenarios for O-RAN virtualized RAN," *White Paper*, 2020.

[5] G.-T. ITU-T, "Transport network support of IMT-2020/5G," 2018.

[6] P. Marsch *et al.*, *5G system design: architectural and functional considerations and long term research*. John Wiley & Sons, 2018.

[7] T. ETSI, "5g;ng-ran;architecture description (3gpp ts 38.401 version 16.5.0 release 16)," *ETSI, Tech*, 2021.

[8] R. Agrawal *et al.*, "Cloud RAN challenges and solutions," *Annals of Telecommunications*, vol. 72, no. 7-8, pp. 387–400, 2017.

[9] 3GPP, "Study on new radio access technology; radio access architecture and interfaces (release 14)," 3rd Generation Partnership Project (3GPP), Technical Recommendation (TR) 38.801, 2017.

[10] A. Garcia-Saavedra and X. Costa-Perez, "O-ran: Disrupting the virtualized ran ecosystem," *IEEE Communications Standards Magazine*, 2021.

[11] P. Sehier *et al.*, "Transport evolution for the RAN of the future," *Journal of Optical Communications and Networking*, vol. 11, no. 4, pp. B97–B108, 2019.

[12] G. N.-P. ETSI, "Network functions virtualisation (NFV): NFV performance & portability best practises," *ETSI, Tech*, 2014.

[13] R. Guerzoni *et al.*, "Network functions virtualisation: an introduction, benefits, enablers, challenges and call for action, introductory white paper," in *SDN and OpenFlow World Congress*, vol. 1, 2012, pp. 5–7.

[14] C. Badulescu and J. Triay, "ETSI NFV, the pillar for cloud ready ict deployments," *Journal of ICT Standardization*, vol. 7, no. 2, pp. 141–156, 2019.

[15] F. W. Murti, A. Garcia-Saavedra, X. Costa-Perez, and G. Iosifidis, "On the optimization of multi-cloud virtualized radio access networks," *arXiv preprint arXiv:2002.10681*, 2020.

[16] F. W. Murti *et al.*, "Optimal Deployment Framework for Multi-Cloud Virtualized Radio Access Networks," *IEEE Transactions on Wireless Communications*, pp. 1–1, 2020.

[17] A. Garcia-Saavedra *et al.*, "Fluidran: Optimized VRAN/MEC orchestration," in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 2018, pp. 2366–2374.

[18] J. Yusupov *et al.*, "Multi-objective function splitting and placement of network slices in 5G mobile networks," in *2018 IEEE Conference on Standards for Communications and Networking (CSCN)*. IEEE, 2018, pp. 1–6.

[19] N. Molner *et al.*, "Optimization of an integrated fronthaul/backhaul network under path and delay constraints," *Ad Hoc Networks*, vol. 83, pp. 41–54, 2019.

[20] S. Matoussi *et al.*, "5G RAN: Functional split orchestration optimization," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 7, pp. 1448–1463, 2020.

[21] L. Mamushiane *et al.*, "Overview of 9 Open-Source Resource Orchestrating ETSI MANO Compliant Implementations: A Brief Survey," in *2019 IEEE 2nd Wireless Africa Conference (WAC)*. IEEE, 2019, pp. 1–7.

[22] V. Q. Rodriguez *et al.*, "Cloud-RAN factory: Instantiating virtualized mobile networks with ONAP," in *2020 11th International Conference on Network of the Future (NoF)*. IEEE, 2020, pp. 120–122.

[23] E. G. N.-I. 029, "Network functions virtualisation (nfv) release 3; architecture; report on the enhancements of the nfv architecture towards "cloud-native" and "paas"," *ETSI, GR*, 2019.

[24] Cloud Native Computing Foundation, "Kubernetes Documentation — Kubernetes," 2020. [Online]. Available: https://kubernetes.io/docs/home/

[25] T. L. Foundation. (2020) Kubernetes Documentation. [Online]. Available: https://kubernetes.io

[26] L. M. P. Larsen, A. Checko, and H. L. Christiansen, "A survey of the functional splits proposed for 5G mobile crosshaul networks," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 1, pp. 146–172, 2019.

[27] L. Gavrilovska, V. Rakovic, and D. Denkovski, "From Cloud RAN to Open RAN," *Wireless Personal Communications*, vol. 113, no. 3, pp. 1523–1539, 2020. [Online]. Available: https://doi.org/10.1007/s11277-020-07231-3

[28] C. J. Bernardos *et al.*, "Network virtualization research challenges," IETF, Tech. Rep., 2019.

[29] A. Rehman *et al.*, "Network functions virtualization: The long road to commercial deployments," *IEEE Access*, vol. 7, pp. 60 439–60 464, 2019.

[30] B. Yi *et al.*, "A comprehensive survey of network function virtualization," *Computer Networks*, vol. 133, pp. 212–262, 2018.

[31] L. Bonati *et al.*, "Open, programmable, and virtualized 5G networks: State-of-the-art and the road ahead," *arXiv preprint arXiv:2005.10027*, 2020.

[32] Free5GC, "Free5GC," https://free5gc.org/, 2021.

[33] O. S. Alliance, "OpenAirInterface," https://www.openairinterface.org/, 2021.

[34] P.-C. Lin and S.-L. Huang, "Performance profiling of cloud radio access networks using openairinterface," in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, 2018, pp. 454–458.

[35] Y. Hu and J. Wang, "Architectural and Cost Implications of the 5G Edge NFV Systems," in *IEEE 37th International Conference on Computer Design (ICCD)*, 2019, pp. 594–603.

[36] S. C. Forum, "Small cell virtualization: Functional splits and use cases," *Small Cell Forum*, 2016.

[37] F. Z. Morais *et al.*, "PlaceRAN: Optimal Placement of Virtualized Network Functions in the Next-generation Radio Access Networks," *XX*, vol. XX, pp. 1–13, 2021. [Online]. Available: http://arxiv.org/abs/2102.13192

[38] L. Foundation. (2020) Open Network Automation Platform. [Online]. Available: https://https://www.onap.org//

[39] E. OSNM. (2020) Open Source MANO. [Online]. Available: https://osm.etsi.org/

[40] N. Nikaein *et al.*, "Mosaic5G: Agile and flexible service platforms for 5G research," *ACM SIGCOMM Computer Communication Review*, vol. 48, no. 3, pp. 29–34, 2018.

[41] N. Makris *et al.*, "Virtualized heterogeneous 5g cloud-ran deployment over redundant wireless links," in *2019 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*. IEEE, 2019, pp. 1–2.

[42] K. Kondepu *et al.*, "Orchestrating lightpath adaptation and flexible functional split to recover virtualized ran connectivity," in *2018 Optical Fiber Communications Conference and Exposition (OFC)*. IEEE, 2018, pp. 1–3.

[43] R. K. Saha *et al.*, "Novel M-CORD based multi-functional split enabled virtualized cloud ran testbed with ideal fronthaul," in *2018 IEEE Globecom Workshops (GC Wkshps)*. IEEE, 2018, pp. 1–7.

[44] L. Seung-Que *et al.*, "Virtualization and orchestration of radio access network," in *2018 IEEE International Conference on Communication Systems (ICCS)*. IEEE, 2018, pp. 321–325.

[45] A. G. Dalla-Costa *et al.*, "Orchestra: A Customizable Split-Aware NFV Orchestrator for Dynamic Cloud Radio Access Networks," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 6, pp. 1014–1024, 2020.

[46] B. Dzogovic *et al.*, "Connecting Remote eNodeB with Containerized 5G C-RANs in OpenStack Cloud," in *2019 6th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2019 5th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom)*. IEEE, 2019, pp. 14–19.

Julio Renner is a Software Developer at Routific, Canada. Graduated in Computer Science at the University of Vale do Rio dos Sinos (UNISINOS), Brazil. His research interests include software containerization and orchestration, DevOps and cloud computing.

Gabriel Matheus F. de Almeida has been a Computer Science Researcher and a member with the Laboratory Computer Networks and Distributed Systems, Federal University of Goiás (UFG), Brazil, since 2019. His research interests include wireless networks, software-defined networks, virtualization, resource allocation and performance evaluation.

Luis M. Contreras earned a Telecom Engineer (M.Sc.) degree at the Universidad Politécnica of Madrid (1997), and holds an M.Sc. on Telematics from the Universidad Carlos III of Madrid (2010). Since August 2011 he is part of Telefónica I+D / Telefónica CTIO unit, working on 5G, SDN, virtualization, and transport networks. He is part-time lecturer at the Universidad Carlos III of Madrid, where is also pursuing a Ph.D. (expected for 2021).

Rodrigo da Rosa Righi is professor and researcher at University of Vale do Rio dos Sinos (UNISINOS), Brazil. Rodrigo concluded his post-doctoral studies at KAIST - Korean Advanced Institute of Science and Technology, South Korea. He obtained his MS and PhD degrees in Computer Science from the Federal University of Rio Grande do Sul, Brazil, in 2005 and 2009, respectively. He is a member of both IEEE Computer Society (senior member) and ACM.

Kleber Vieira Cardoso is an associate professor at the Institute of Informatics – Universidade Federal de Goiás (UFG), where he has been a professor and researcher since 2009. He holds a degree in Computer Science from UFG (1997), has MSc (2002) and PhD (2009) in Electrical Engineering from COPPE – Universidade Federal do Rio de Janeiro. In 2015, he spent his sabbatical at Virginia Tech (in the USA) and, in 2020, at Inria Saclay Research Centre (in France).

Fernando Zanferrari Morais is MSc degree at University of Vale do Rio dos Sinos (UNISINOS), Brazil. Fernando received the bachelor's degree from Centro Universitário Lasalle in 2009. He served as a Telecommunication Engineer at Vivo Operator, a Telefonica group, Brazil. His research interests include telecommunication mobile networks, software-defined networking, virtualization and cloud computing.

Cristiano Bonato Both is a professor of the Applied Computing Graduate Program at the University of Vale do Rio dos Sinos (UNISINOS), Brazil. He coordinators research projects funded by H2020 EU-Brazil, CNPq, FAPERGS, and RNP. His research focuses on wireless networks, next-generation networks, softwarization and virtualization technologies for telecommunication networks.

Gustavo Zanatta Bruno is a Ph.D. student in Computer Science at the University of Vale do Rio dos Sinos (UNISINOS) and MSc in Computer Science at Universidade Federal Fluminense (UFF), both in Brazil. He served as an Information Technology Infrastructure Analyst at MTI, a public company responsible for IT services in Mato Grosso, Brazil. His research interests include infrastructure automation, mobile telecommunication networks, and cloud computing.