AIRTIME: End-to-end Virtualization Layer for RAN-as-a-Service in Future Multi-Service Mobile Networks

Maicon Kist, João F. Santos, Diarmuid Collins,
Juergen Rochol, Luiz A. DaSilva, and Cristiano B. Both

Abstract—Future mobile networks are envisioned to become multi-service systems, enabling the dynamic deployment of services with vastly different performance requirements, accommodating the needs of diverse service providers. Virtualizing the mobile network infrastructure is of fundamental importance for realizing this vision in a cost-effective manner. While there have been extensive research efforts in virtualization for the mobile core network, virtualization in the radio access network (RAN) is still at an early stage. In this paper, we present AIRTIME, a new RAN slicing system that enables the dynamic on-the-fly virtualization of RANs, with the programmability required by service providers to customize any aspect of their virtual RAN to meet their service needs. We present a prototype implementation of AIRTIME and evaluate the: (*i*) capacity to create virtual RANs on-the-fly, (*ii*) performance experienced by slice owners, (*iii*) isolation among multiple virtual RANs sharing the same physical infrastructure, and (*iv*) scalability to accommodate a large number of virtual RANs.

Index Terms—Virtualization, Software-Defined Radio, Software-Defined Networking, Network Function Virtualization, RAN as a Service, Network Slicing

1 Introduction

The fifth generation of mobile networks (5G) is expected to significantly scale up and improve the cellular network to support a wide range of use-cases and provide services tailored to satisfy simultaneously various customer demands and device types [1]. This *service-oriented* vision presents significant opportunities for the network architecture, with benefits such as ease in service creation, network sharing, reduced CAPEX/OPEX, enhanced user experience, and energy efficiency [1].

The International Communications Union (ITU) identifies three classes of services in 5G: enhanced Mobile BroadBand Communications (eMBBC), Ultra-Reliable and Low-Latency Communication (URLLC), and Massive Machine Type Communications (mMTC), and several specific services within these classes [2]. A one-size-fits-all architecture is unlikely to be suitable for such diverse services. Realizing the service-oriented 5G vision in a cost-effective manner requires a flexible mobile network architecture that can turn the physical network infrastructure into multiple logical end-to-end slices, one for each service [3], [4]. Each slice in such an architecture is a complete virtualized network instance, spanning both the Core Network (CN) and the Radio Access Network (RAN), which is tailored to meet the requirements of the service being provided. The most prominent 5G architectures are built upon this idea of slicing the mobile network infrastructure [5], [6], [7], [8], [9], [10]. In recent years, frameworks for Software-Defined Network (SDN) and Network Function Virtualization (NFV), two key enablers for slicing in the network, have become increasingly popular and full-featured. Research prototypes leverage both paradigms to present advanced state-of-the-art architectures for Software-Defined RAN (SD-RAN) [11], [12], CN slicing [9], [13], [14], or specialized CNs for specific types of end-users [15].

The concept of RAN-as-a-Service (RANaaS) requires the ability to dynamically create, manage, and configure virtual RANs, which can be tailored to meet the requirements of an end-to-end service. RAN slicing is a new and challenging problem that is only starting to receive attention. The realization of RANaaS poses three key challenges:

- **slice isolation** so that the operation of one slice does not interfere with the operation of other slices coexisting over the shared physical infrastructure or spectrum,
- **slice programmability** so that Service Providers (SPs) can tailor any aspect of their virtual RAN to meet the respective service requirements, without violating the first objective, and
- **slice adaptability** so that infrastructure owners can dynamically change the resources allocated to a given slice (radio, baseband processing, and fronthaul bandwidth) to increase resource usage efficiency, without violating the first objective, and to further enhance the second objective.

The current state-of-the-art solutions address only part of these objectives. More precisely, the initial efforts in

- *Maicon Kist and Juergen Rochol are with the Department Computer Science, Federal University of Rio Grande do Sul, Porto Alegre, RS, Brazil. Email: {mkist, juergen}@inf.ufrgs.br*
- *João F. Santos, Diarmuid Collins, and Luiz A. DaSilva are with CONNECT, Trinity College Dublin, Ireland. Email: facocalj, collindi, dasilval@tcd.ie*
- *Cristiano B. Both is with the Applied Computing Graduate Program at University of Vale do Rio dos Sinos (UNISINOS), Brazil. Email: cbboth@unisinos.br*

this area have focused on the efficient scheduling of radio resources and, as a consequence, constrained all slices to the same CN/RAN of the underlying physical infrastructure [5], [6]. Recent advancements have added some level of programmability in the radio base station [14], [16], [17]. However, these works do not consider an end-to-end RAN slicing system that fully addresses all aforementioned challenges.

In this article, we present AIRTIME, the first RAN virtualization system that provides end-to-end programmability, while ensuring isolation and enabling flexible and adaptive provisioning of RAN resources to virtual RANS (vRANs) based on their requirements. Each vRAN in AIRTIME is an end-to-end RAN running on top of a shared physical network infrastructure, with customizable data and control planes based on the service requirements. Our system combines the application of slicing in two levels: (*i*) the virtualized baseband processing unit (vBBU) execution layer and (*ii*) the Remote Radio Head (RRH) hypervisor. The former is responsible for slicing the computing resources in isolated containers, enabling multiple vBBUs to coexist on top of shared processing resources, while the latter acts in the RRH, effectively slicing the radio resources (spectrum and RF front-end) via a new hypervisor introduced in this work. Moreover, the design of AIRTIME allows SPs to flexibly employ the Radio Access Technology (RAT) (a combination of vBBU and radio resources) that best fits the application service requirements. In summary, this work makes the following contributions:

- To present a feasible RAN slicing design that enables isolation among slices and supports the programmability required by SPs to compose a RAN tailored to the service in question;
- To introduce a new set of abstractions for the virtualization of radio resources which can be applied to future mobile network RANs; and
- To implement and validate a prototype of AIRTIME on a reconfigurable radio testbed.

The remainder of this article is structured as follows. In Section 2, we present the evolution of radio hardware and its impact towards the virtualization of the RAN. In Section 3, we describe AIRTIME, detailing its BaseBand Unit (BBU) virtualization and RF front-end slicing capabilities. In Section 4, we present the prototype of AIRTIME, describing the implementation details of the `RRH` and `BBU Hypervisors`. In Section 5, we present the experimental results that validate the operation of AIRTIME. In Section 6, we discuss the related work on prototypes of RAN slicing for mobile networks. Finally, Section 7 summarizes our main conclusions and addresses areas for future work.

2 From physical to virtual base stations: Towards RANaaS

In this section, we describe the evolution of base stations and their impact on the design of mobile network architectures. First, we describe the internal architecture of a generic physical base station, highlighting its two key components: the baseband processing and the RF front-end. Then, we present the evolution towards virtualized base stations.

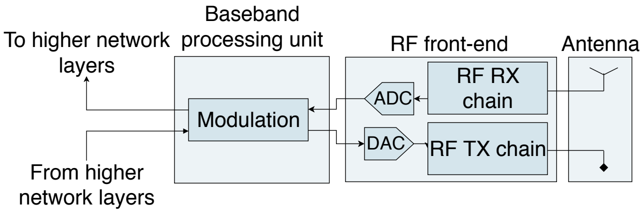


Fig. 1: General architecture of a base station.

2.1 Base Station Architecture

A base station comprises three major components: the BBU, the RF front-end, and the antenna(s). The BBU is responsible for implementing a particular RAT (modulation, frame structure, error correction, and so forth). In modern mobile systems, the BBU is a hardware or software component, and the RF front-end mainly contains analog circuitry, as shown in Fig. 1. The Analog-to-Digital Converter (ADC) and Digital-to-Analog Converter (DAC) are responsible for the interface between the digital domain of the BBU and the analog domain of the RF front-end. The antenna module is located in close proximity of the RF front-end (*e.g.*, a few meters), due to high losses associated with coaxial cables. In the first generation of mobile networks, both the BBU and the Radio Frequency (RF) front-end were coupled in a hardware box that implemented everything that the mobile access technology required. In recent years, the RF front-end has started to be decoupled from the BBU; this enabled mobile operators to replace the BBU module when needed without too much of an effort. Arguably, the main architectural element of the RAN that offers opportunities for virtualization is the base station. And within the base station, virtualization can be performed in the BBU, RRH, or a combination thereof (we explore BBU and RRH virtualization later in this section).

As shown in Fig. 2, the BBU component translates information bits into digital baseband waveforms, and vice-versa. This function is also referred to as digital modulation. It maps a binary sequence to digital waveform samples known as symbols. At the receiver side, the symbols are demodulated to retrieve the embedded binary information. As baseband signals are not suitable for direct transmission, the RF front-end must convert the digital baseband samples into high-frequency analog radio signals. When receiving, the RF front-end selects the desired radio frequency signals, down-converts them to the baseband frequency, and digitizes them into digital In-phase & Quadrature (IQ) samples. In current radio systems, *e.g.*, LTE and WiFi, the radio frequency is divided into fixed-size channels and at a pre-determined central frequency. The baseband signal may be mapped to any pre-defined channel.

2.2 Towards a Centralized RAN

The high-bandwidth data exchange between the BBU and the RF front-end requires a high-bandwidth bus connecting these two domains. This constraint has been removed with the widespread deployment of fiber-optics cables, with the added benefit that the two elements can be located farther away from each other. In the 3G/4G era, baseband processing was implemented on BBUs, with

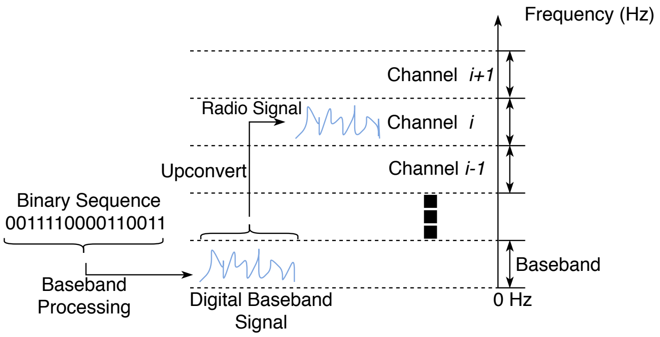
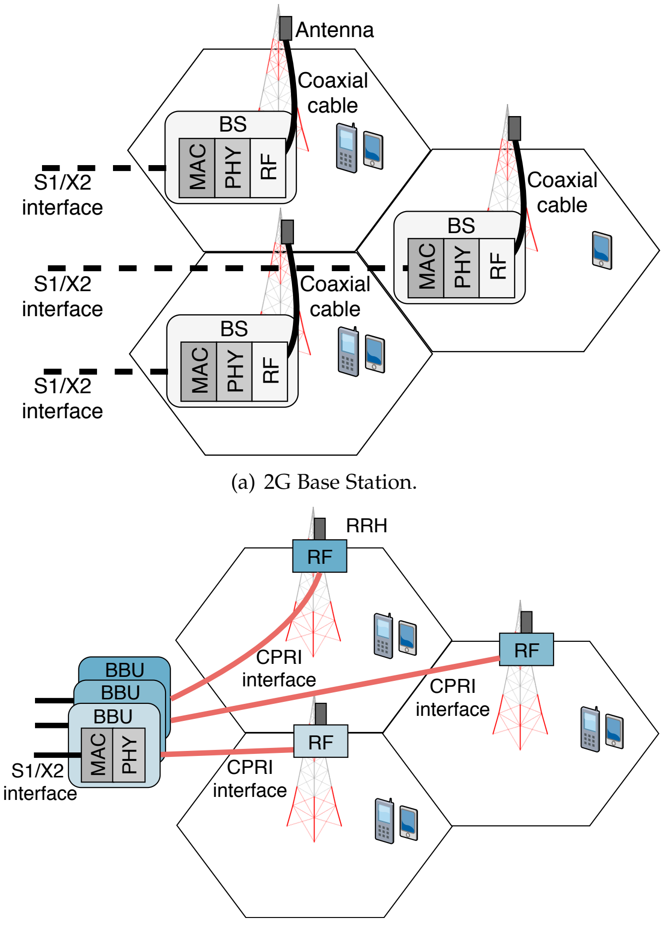


Fig. 2: Transformation from digital data to analog signal.



(a) 2G Base Station.

(b) 3G/4G Base Station with RRH and BBU.

Fig. 3: Base Station architecture evolution.

specialized hardware that implements a RAT, while an RRH integrates the RF front-end and the antenna. In the downlink, *i.e.*, from BBU to RRH, the “data” generated by the BBU is a stream of IQ samples that represent the radio signal that must be transmitted by the RRH. Similarly, in the uplink, *i.e.*, from RRH to BBU, the “data” received by the BBU is the digitized version of the signal received in the RRH. Fig. 3(a) and 3(b) show the evolution from the 2G to the 3G/4G network architecture.

The Cloud-Radio Access Network (C-RAN) leverages the advancements of fiber-optics cables to incorporate cloud-based architectures into the mobile network. In this network architecture, the BBU can be placed in a data center, enabling cost savings through centralized maintenance, cloud facility rental, and reduced environmental impact. The fronthaul network connects the data centers with RRHs, which can be geographically separated by up to 40 Km (the distance limitation comes from the delay constraints imposed by the mobile network protocol stack [18]). Due to the more considerable distances between BBUs and RRHs, it is necessary to adopt a fronthaul network protocol that provides constant bitrate, accurate synchronization and stringent latency control, such as the popular Common Public Radio Interface (CPRI), Open Base Station Architecture Initiative (OBSAI), and Open Radio equipment Interface (ORI).

The deployments of BBUs in a centralized cloud require high-capacity fronthaul links, typically based on optical technology, for transferring the IQ samples to/from RRHs. High bandwidth requirements for the fronthaul have motivated solutions that consider moving some of the baseband processing functions back to the RRH, while the remaining are kept in the BBU. Which of these functions to centralize and which to move closer to the RRH is a decision based on the constraints imposed by the fronthaul capacity and latency requirements of the service provided. Fig. 4 provides an overview of the six main configurations possible (known in the literature as functional splits options), described below (the readers can find a detailed overview of each option in Kist *et al.* [19]):

- In *IQ Forwarding*, samples are transported over the fronthaul to the cloud data center, which centralizes all BBU functions.
- In *Subframe Forwarding*, the CP Removal and Fast Fourier Transform (FFT) functions are moved to the RRH. In this case, only the IQ samples of useful sub-carriers are transported over the fronthaul. Eliminating this overhead reduces the fronthaul bandwidth required [20].
- In *RX Data Forwarding*, the Resource Element (RE) de-mapper is moved closer to the RRH. In this distribution option, the data center receives the IQ samples of REs allocated to mobile users, which is something that can change in each LTE frame. Because of this, the fronthaul data rate required is a fraction of the data rate needed for the Subframe split.
- In *SoftBit Forwarding*, the RRH executes all functions required to recover bits from the radio signal, which includes both user data and higher layer control data, such as MAC headers. This split option reduces the fronthaul data rate required by a fraction of the number of antennas in the RRH, when compared to the previous split for the same RE allocation rate.
- In *MAC Forwarding*, the MAC packet data units are transported over the fronthaul (as is done in 3G/4G mobile networks).

The move towards centralization enables software implementations of all the BBU functions, running on top of high-performance General Purpose Processors (GPPs). Through software upgrades or re-configuration, pro-

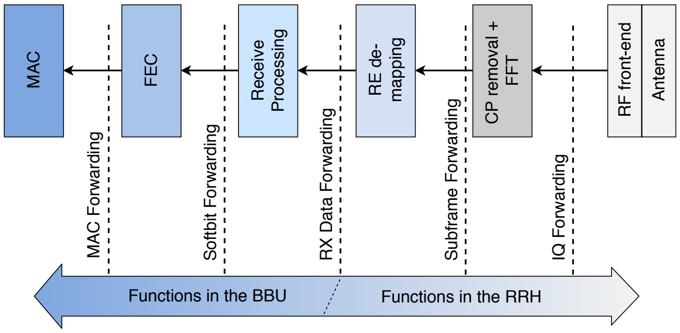
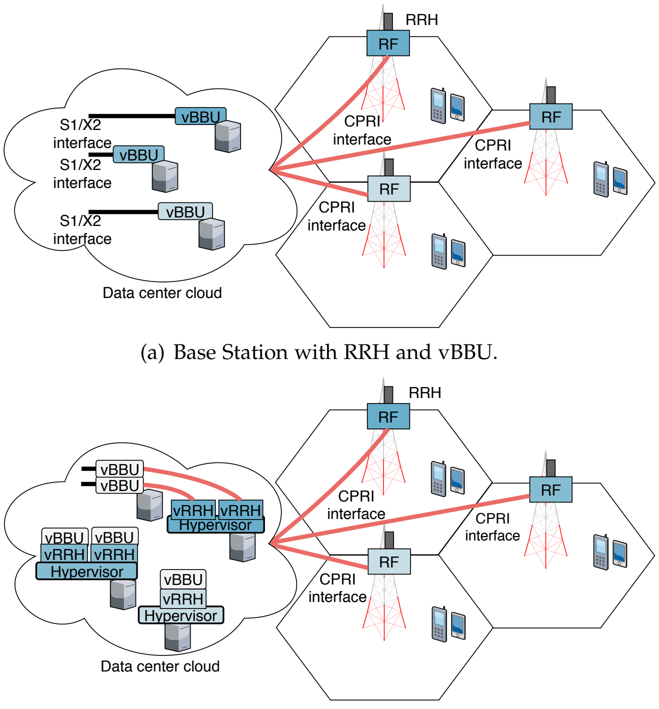


Fig. 4: Functional splits between BBU and RRH.



(a) Base Station with RRH and vBBU.

(b) Base Station with vRRH and vBBU.

Fig. 5: Base Station architecture in RANaaS.

grammable BBUs can implement any RAT (although constraints in the fronthaul bandwidth and latency may hinder the adoption of some).

2.3 Towards RAN as a Service

A natural evolution from the centralized RAN architecture is towards the RANaaS. This new architecture leverages cloud data centers and base station virtualization (a combination of a virtual BBU and a virtual RRH [21]) to dynamically create vRANs on top of the physical infrastructure, each one tailored to meet the requirements of a particular end-to-end service. It is important to emphasize that in this article we consider virtualization as the process in which *physical wireless resources* can be abstracted into virtual resources holding a subset of functionalities of the underlying physical counterpart and shared by ensuring complete isolation from each other. Thus, RAN virtualization is the process of abstracting all elements of the RAN and slicing them into virtual elements holding certain corresponding functionalities and isolated from each other [22].

By offering the capabilities to support multiple vRANs, RANaaS opens up new business models in which SPs can lease vRANs from the infrastructure providers [8]. In this scenario, the infrastructure provider controls all physical resources, comprising the radio spectrum, physical RRH, hardware resources in data centers (*i.e.*, servers with processing, memory, and storage), and the physical network. An SP enters into a contract with the infrastructure provider for one or more vRANs, which include at least one virtual base station, *i.e.*, a Virtual Remote Radio Head (vRRH) attached to a Virtual BaseBand Unit (vBBU). Note that RANaaS is different from the legacy RAN sharing model, in which the focus is only on the sharing of resources among mobile operators, *e.g.*, radio spectrum, network function and application, and baseband processing.

Isolation, programmability, and adaptability are the key elements for enabling end-to-end vRAN customization to accommodate the different services envisioned for 5G mobile networks. These three elements constitute the foundation for a multi-service and multi-tenant RANaaS [23], which is realized by applying the principles of base station virtualization.

Virtualization can be implemented in a BBU to allow multiple vBBUs to run on top of the same physical hardware, or in an RRH to allow multiple vRRHs to run on top of physical RRHs. In this article, we argue for a combination of BBU and RRH slicing to instantiate complete end-to-end virtual RANs on top of the physical infrastructure.

2.3.1 *Virtual BBUs*

A vBBU implements the signal processing operations to transform a sequence of bits into IQ samples that represent the radio signal, which must be transmitted (exactly like the physical hardware-based version does). A vBBU pool can be executed on a GPP, leveraging highly-optimized signal processing libraries, as well as taking advantage of the ever-increasing evolution of processors, such as higher processing power and energy efficiency. Fig. 5(a) illustrates this architectural evolution.

The first research efforts in BBU virtualization considered a static functional split between vBBU and RRH, in which the functions in RRH are still implemented on specialized hardware, while the remaining functions are moved to the vBBU [19]. Recently, the 3GPP RAN3 working group has considered a vBBU split consisting of two new entities, named Distributed Unit (DU) and Central Unit (CU). The former can host time-critical functions of the physical layer, whereas the latter hosts non-time critical features, such as the upper Medium Access Control (MAC) layer. The DU is envisioned to cover a 10-20 Km radius, while the CU covers 100-200 Km [24].

We have presented in [16] a similar approach to 3GPP RAN3, in which we propose the fine-grained virtualization of the baseband functions. More precisely, each of the baseband functions is realized as an independent Virtual Network Function (VNF) that can be migrated on-the-fly between regional and edge data centers. The regional and edge data centers share similarities with the DU and CU, in the sense that each hosts baseband functions with different requirements. This flexible functional split can greatly impact the performance of RAN slicing, and the optimal split depends mainly on the characteristics of the target service. For example, URLLC vRANs may require most of the BBU functions to run on the edge data center, to cope with the latency requirements, while in an eMBBC vRAN, a higher centralization can enhance the end-user throughput by aggregating RRHs [25].

2.3.2 Virtual RRH

RRH virtualization is a subset of base station virtualization, in which a physical RRH is abstracted into vRRHs holding certain corresponding functionalities of their physical counterpart. As mentioned previously, a physical RRH is responsible for translating the stream of IQ samples generated by BBUs into over-the-air radio signals and converting over-the-air radio signals received by the antenna into a stream of IQ samples that are sent to the (v)BBU. In summary, a vRRH must operate exactly like its physical counterpart and a (v)BBU should not be able to distinguish between a physical or virtual RRH.

In contrast to BBU virtualization, which can employ conventional cloud computing technologies, the virtualization of RRHs is still in the early stages of development. Applying virtual machines or container-based solutions in this domain does not adequately address the problem, as they do not deal with the additional dimension of virtualizing and isolating radio resources (spectrum and radio hardware). The existing RRH virtualization approaches that account for the slicing of radio resources fall into one of two categories:

- Slice and assign high-level radio resources between vRRH instances by employing a common physical and lower MAC layers [9], [26].
- Slice and assign chunks of spectrum between vRRHs, which then interact with a (v)BBU [27].

The virtualization of high-level radio resources constrain all virtual counterparts to use the same physical and MAC layers and is a way of using well-known resource sharing mechanisms to enable RRH virtualization. This approach has been adopted by several research efforts that focused on virtualizing WiFi access points [28], [29], WiMAX [5], [30] and 4G base stations [31], [32], [33]. However, this approach goes against the main principles advocated for RANaaS deployments, *i.e.,* flexibility, programmability, and adaptability, as all vRANs must share the same the RAT (physical and MAC layer), hindering the specialization towards particular services.

The state-of-the-art in RRH virtualization uses a radio hypervisor for the slicing of low-level radio resources and for provisioning a dedicated chunk of radio resources for each vRRH, which then interfaces with a (v)BBU, as shown in Fig. 5(b). Arguably, this is the best approach to enable RANaaS, since it allows vRRHs to adopt any RAT. Although a radio hypervisor that operates using low-level radio resources is more flexible, programmable and adaptable, only recently prototypes started to be developed [16], [17]. In the next section, we present AIRTIME, a RANaaS system that enables SPs to flexibly customize their RATs to best fit their requirements.

Although several challenges in C-RANs are investigated and solutions based on virtualization are presented, we focus our attention on the following gaps identified in the state-of-the-art: (*i*) End-to-end network services are still designed to support a single "one-size-fits-all" RAN and RAT. These services are not adequate for future mobile networks without a more thorough re-design. (*ii*) Virtualization solutions adopt an atomic vBBU approach in which all of the baseband processing is realized in one data center, while RRHs incorporate only the RF front-end. This fixed distribution of functionalities significantly degrades the coverage area of RAN, as the maximum distance between the data center and RRH is limited due to latency constraints. (*iii*) The C-RAN architecture places a tremendous bandwidth requirement on the fronthaul network, in practice constraining the fronthaul to be composed solely of optical links. A proper solution that enables the adoption of heterogeneous fronthaul links can significantly accelerate the adoption of this architecture in future mobile networks. These challenges must be addressed in the current state-of-the-art mobile network architectures so satisfy all 5G requirements. In the next section, we present AIRTIME, an architecture that integrates RRH and BBU virtualization to enable all 5G requirements.

3 AIRTIME Architecture

The main contribution of this article is the integration of BBU and RRH virtualization to realize the requirements of a multi-service RANaaS architecture: *isolation* among multiple SPs to achieve the *flexibility* to customize and control any aspect of the vRAN and the *adaptability* to allocate physical resources for efficient resource usage. Our system can support running multiple vRANs on top of the physical infrastructure and tailoring each vRAN to a service. To this end, the design of AIRTIME explicitly separates the infrastructure provider and SPs, as shown in Fig. 6. In the remainder of this section we detail the operation of AIRTIME, as well as the `BBU Hypervisor` and the new `RRH Hypervisor`.

In AIRTIME, the RAT of each vRAN can be tailored to the particular requirements of the SP by adjusting the radio resources in a vRRH, and especially the baseband functions in the vBBU. At the vRRH, the SP can tailor the spectrum bandwidth to better accommodate its users. It is in the vBBU that enormous opportunities for optimization occur in the form of changing baseband parameters or replacing the entire baseband function VNF to satisfy the service requirements (we will present more details on this in Section 4).

The physical RAN encompasses the elements found in future mobile network architectures [21]: (*i*) the high-capacity CU data centers for performing the non-time critical functions for all vBBUs, (*ii*) the low-capacity DU data center that hosts time-critical functions of vBBUs that are bound to a physical RRH within a 10-20 Km radius, and (*iii*) a fronthaul network that interconnects CUs, DUs, and RRHs. Having the DU closer to the RRH supports moving baseband processing VNFs closer to the RRH to reduce bandwidth and latency requirements over the fronthaul network.

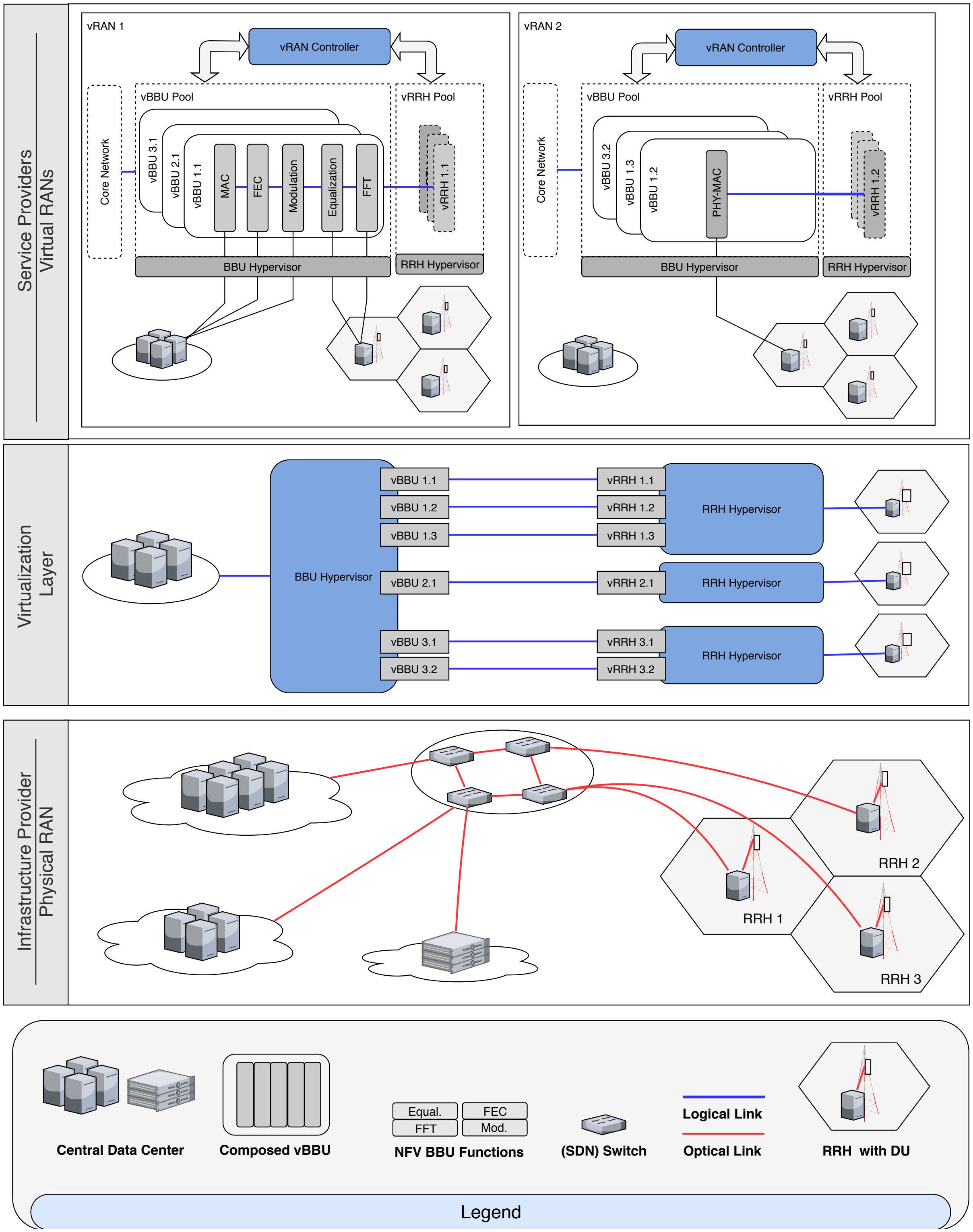


Fig. 6: High-level overview of AIRTIME.

The virtualization layer is the pillar of AIRTIME design, as it contains the `BBU` and `RRH Hypervisor` that sits over the physical infrastructure. This layer is responsible for managing the vRANs, for ensuring complete isolation of radio, processing, and networking resources (radio resource isolation to guarantee that each vRANs can adopt any RAT, processing resource isolation to ensure performance stability, and networking isolation to ensure that changes in one vRAN configuration are self-contained). Primarily, the virtualization layer binds multiple isolated vRANs to the physical infrastructure and provides them with a virtual view of the underlying resources allocated to them. This layer must also offer a set of interfaces for managing the vRANs and a set of interfaces to map these changes to the physical resources.

SPs, such as Mobile Virtual Network Operators (MVNOs) and vertical markets (such as on-demand video streaming, Internet-of-Things for smart cities, and factory automation), realize their vRAN through the creation of virtual base stations over the virtualization layer. Each virtual base station is a composition of a vBBU (which in turn is a chain of baseband VNFs) and a vRRH, all of them managed by an independent `vRAN Controller`. In the next subsections, we present the main elements of the virtualization layer and the `BBU` and `RRH Hypervisor`, as well as the `vRAN Controller`.

3.1 BBU Hypervisor

The `BBU Hypervisor` provides the framework for the execution of baseband processing VNFs and their abstraction into a vBBU. The main benefit of this approach is the small virtualization footprint, while simultaneously enabling the execution of baseband functions in heterogeneous hardware devices.

The `BBU Hypervisor` follows the ETSI NFV model to abstract the physical infrastructure of the CU/DU data centers. This model already considers a virtualization architecture that comprises a set of VNFs that are deployed over the physical infrastructure and that are interconnected. The different arrangements of the set of VNFs in the physical infrastructure give rise to different composition options, each one with its processing, storage, and networking requirements, *e.g.,* deploying the entire vBBU into a single physical machine requires more resources than deploying only a subset of it as VNFs. Moreover, a `BBU Hypervisor` maps VNFs to dedicated virtualization containers (*e.g.,* virtual machine or Linux container) or even physical resources that do not support virtualization (*e.g.,* FPGAs or DSPs).

The workload of a vBBU is highly dependent on the access technology it is implementing. An LTE vBBU, for example, implements the encryption/decryption process of the Packet Data Convergent Protocol (PDCP), the concatenation of the Radio Link Control (RLC) protocol, the radio scheduling and Hybrid Automatic Repeat-reQuest (HARQ) management handled by the MAC layer, and all the baseband processing functions of the physical layer: FEC, modulation/demodulation, equalization, FFT and Cyclic Prefix (CP) addition. Packets from the CN arrive at the vBBU and are sequentially handled by each of the VNFs in sequence before being presented in the form of IQ samples at the vRRH. The VNFs' run-times are independent of each other and directly relate to the amount of data that comes to/from each RRH.

VNFs implementing baseband processing require a large amount of computational capacity. Topping the list are FEC and receive processing VNFs, which together add up to more than half of the computational capacity required by a BBU, in both the downlink and uplink. Different from functions closer to the vRRH, the computational requirements of FEC and receive processing are highly dependent on the radio channel conditions, because a channel with low SNR requires more redundancy for error correction, which translates into more computational time to perform the encoding and decoding tasks.

Running a vBBU as a composition of baseband VNFs enables a higher level of programmability for the SPs. SPs can tailor any aspect of the vBBU by changing the flowgraph of VNFs, *e.g.,* by adding a baseband VNF that performs carrier aggregation to the chain of functions that build vBBUs. Computationally intensive baseband VNFs can be migrated to high-performance and non-virtualizable FPGAs and DSPs to achieve higher data throughput, at the cost of less flexibility. Similarly, different vRANs could employ different baseband processing VNF setups, optimized for their particular service. The flexible placement of VNFs, coupled with RRH virtualization, allows the creation of flexible vRANs adapted to SPs' needs.

3.2 RRH Hypervisor

An `RRH Hypervisor` that slices the radio resources is one of the main novelties of AIRTIME. It is responsible for creating the virtualized version of the physical RRH, *i.e.,* vRRHs, for ensuring their complete isolation, and for facilitating efficient sharing of the underlying physical radio resources. Essentially, the RRH Hypervisor for a RANaaS system should: (*i*) abstract the physical RRH by adding a layer of indirection that maps physical radio resources to a vRRH, providing a virtual view of underlying radio resources, (*ii*) enable vRANs to adopt any vBBU (*i.e.,* any RAT), and (*iii*) apply changes in the vRRH by mapping them to the physical RRH, ensuring that these changes do not interfere with other coexisting vRRHs. The `RRH Hypervisor` is an essential part of the infrastructure provider software framework for supporting RANaaS.

The `RRH Hypervisor` must be logically located between the physical RRH and vBBU and acts as an intermediate layer between these two components. Here, the vBBU interacts solely with a vRRH and does not have any access to its physical counterpart. The vBBU operates as if it were connected to a physical RRH, *i.e.,* generating IQ samples in the downlink and receiving IQ samples in the uplink. In the downlink, the `RRH Hypervisor` intercepts the IQ samples from multiple vBBUs and then reshapes them into a stream of IQ samples that contain the multiplexed signals of all vBBUs. In the uplink, the hypervisor receives the IQ samples of a wideband chunk of spectrum that encompasses the spectrum of all vRRHs. The hypervisor then forwards to the vBBUs only the IQ samples of the portion of the spectrum corresponding to the respective vRRH. Moreover, a vRRH must provide the same set of configuration options as the physical RRH, such as a channel in use, bandwidth, and transmission and reception gain.

The `RRH Hypervisor` should be platform-agnostic to enable vRANs to adopt any vBBU or combination of baseband processing VNFs. This slicing process is a challenging task, as different access technologies have different abstractions for the underlying wireless spectrum. For example, some access technologies divide the full radio spectrum bandwidth into smaller chunks of transmission with a constant duration (such as LTE and WiMAX physical resources blocks), while others allocate the entire bandwidth to users and change the transmission interval according to the traffic demands (such as WiFi and LoRA transmission frames). The `RRH Hypervisor` should not use any of these abstractions, otherwise it would make the slicing mechanism overly complicated, especially if it needs to operate with multiple abstractions simultaneously.

Configurations performed in the vRRH must be mapped to internal settings of the `RRH Hypervisor` or configurations in the physical RRH. The hypervisor should guarantee that any setting performed for one vRRH does not cause any change in other vRRHs. This guarantee is one of the fundamental features that any virtualization layer must adhere to, and the complexity of such a task is related to the internal slicing mechanism used by the `RRH Hypervisor`. The fact that the `RRH Hypervisor` handles only IQ samples significantly eases the fulfillment of this requirement because most, if not all, configurations can be translated to modifications in the stream of IQ samples of the vRRH.

3.3 vRAN Controller

The `vRAN Controller` is a logical entity that gives SPs the ability to manage and control their set of vRRHs and vBBUs in a way that best fits their application requirements. It runs as a high-level orchestrator, responsible for tailoring the functionality and managing the allocation of resources to applications and users associated with the SP's vRAN as if it were operating using dedicated infrastructure. In this sense, the `vRAN Controller` is similar to an SDN Controller, but with a focus on optimizing the wireless interface between mobile users. It is important to highlight that this proposal does not have a focus on the heuristics or algorithms to optimize the assignment and use of physical resources to vRANs. However, AIRTIME offers all interfaces to perform such optimizations.

We can define three Application Programming Interfaces (APIs) for the `vRAN Controller`: (*i*) northbound API, (*ii*) southbound API, and (*iii*) eastbound API. The northbound API is used by SPs' applications to request the tailoring of high-level requirements (such as throughput, latency, and packet loss) to a particular set of users. The southbound interface, in turn, is used by the `vRAN Controller` to interact with the vBBUs and vRRH to (*i*) tailor the virtual infrastructure so that the requirements of the application are met and (*ii*) react to events occurring in the physical infrastructure that propagate into changes in the virtual elements. The eastbound interface is used to interact with the infrastructure provider to manage the life-cycle of all virtual components, *e.g.,* creation, installation, migration of a specific VNF, or entire vBBUs.

The `vRAN Controller` is also responsible for implementing the control protocols required for the communication and coordination of vRRHs and vBBU with the rest of the mobile infrastructure (*e.g.,* S1 and X2 interfaces in LTE). This communication means that all operations defined for a given RAN architecture can be supported so long as the appropriate interfaces and messages are implemented as part of the respective `vRAN Controller`. The communication of `vRAN Controllers` with `BBU` and `RRH Hypervisor` is message-based and shares the same physical network with data traffic, *i.e.,* the IQ samples between baseband processing VNFs and/or vRRH. This allows the `vRAN Controller` to be deployed as a VNF on top of the same physical infrastructure of the hypervisors or even in different CU/DU data centers. Moreover, it gives the SPs flexibility to design their management system with varying levels of centralization based on their service needs and allows coordination among vRANs under the same SP ownership.

The BBU and RRH virtualization capabilities, allied with the distribution of baseband VNF within the physical infrastructure, enables AIRTIME to offer a multi-service and multi-tenant RAN. Slice owners can tailor their vRAN slice to the requirements of their services by changing the baseband processing VNF according to their needs, *i.e.,* similarly to what is done in current baseband processing functions implemented in Software-Defined Radio platforms [34], or by adding or removing a baseband processing VNF from the vBBU chain. The CU/DU data centers play a major role in enabling latency-stringent services and in reducing the fronthaul bandwidth requirements by allowing the execution of vBBU baseband functions closer to the mobile subscriber. For example, all baseband processing VNFs can be moved to the DU center if low bandwidth or high latency links are connecting the DU and the CU. In the next section we present a prototype of AIRTIME that implements the functionalities described so far.

4 AIRTIME Design and Implementation

In this section, we present the system design for both the `BBU Hypervisor` on general-purpose machines and the `RRH Hypervisor` on a Software-Defined Radio (SDR) platform. We have kept the discussion more general for the `BBU Hypervisor`, as its implementation relies on existing and ever-evolving frameworks for SDN and NFV. We present the `RRH Hypervisor` in greater detail, as it is one of the main innovations introduced in AIRTIME. Fig. 7 gives an overview of the major components of our prototype, which we discuss in the remainder of this section.

4.1 BBU Hypervisor

The `BBU Hypervisor` comprises four top-level components: Open Source Mano (OSM), OpenStack, LXC, and ONOS. OSM (Release 4 - R4) provides the functionality that enables (*i*) requesting the creation or configuration of a vRAN slice thorough the OSM client API, (*ii*) providing a database of VNFs and vBBUs that can be deployed in the physical infrastructure, (*iii*) managing the life-cycle of all virtual resources, including vRAN, vBBU, and baseband processing VNFs, and (*iv*) interfacing with OpenStack, ONOS, and the `RRH Hypervisor` to deploy an end-to-end slice over the physical infrastructure. OpenStack is the virtual infrastructure manager that bridges OSM with the three types of physical resources of the data centers, RRHs, and the SDN network interconnecting both. In the data centers, we use LXC, the light-weight container-based virtualization system that allows running baseband processing VNFs on the data center computers; and ONOS is used to configure the flow tables of SDN switches, so that the high-level flowgraph of the vBBU is embedded in the physical network.

SPs request the creation or configuration of a vRAN slice by sending a JSON data object to the OSM client REST API.

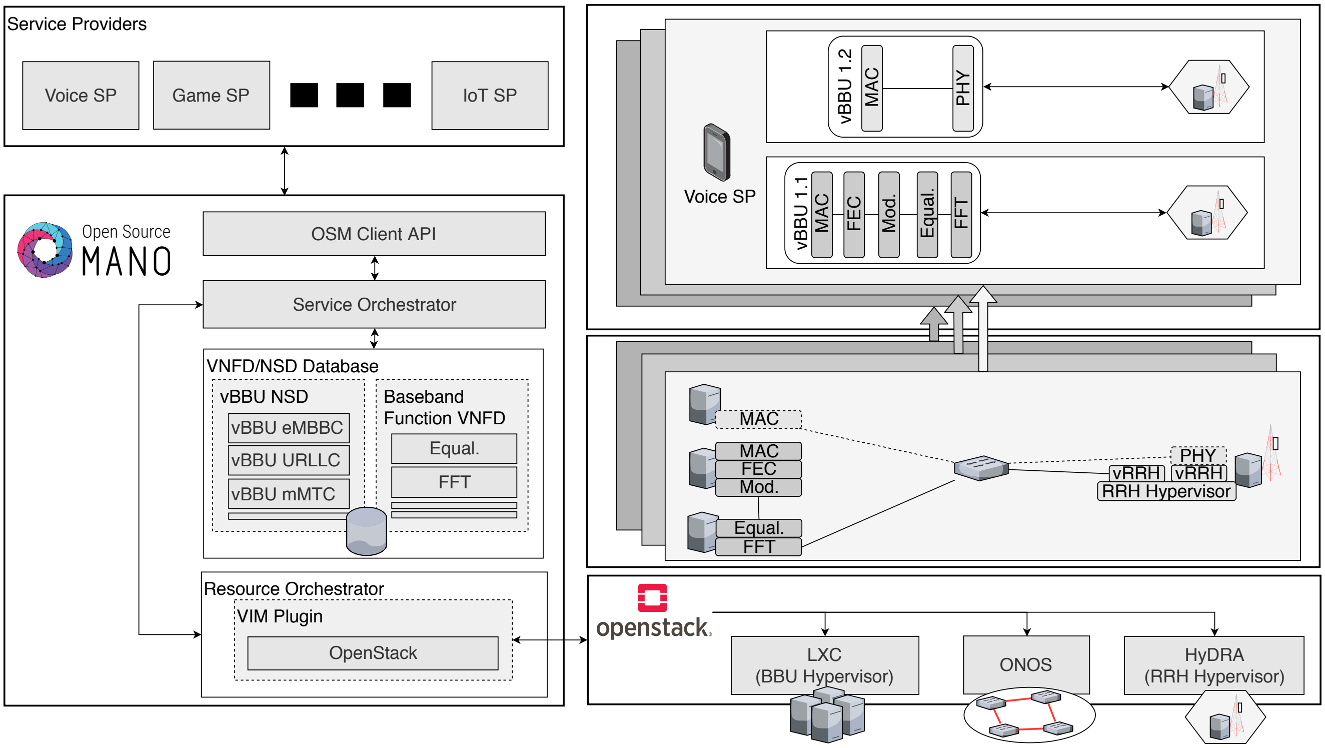


Fig. 7: Design and implementation.

The JSON message specifies the physical RRH in which the SP wants over-the-air coverage and the vBBU that must be connected to the soon-to-be-created vRRH on top of the selected physical RRH. Specifying the vBBU can be done in one of two ways: in the first option, the SP can use one of the available vBBUs in the OSM database, such as eMBBC, URLLC, or mMTC; in the second option, the SP can specify a set of baseband VNFs (also from the OSM database) that together form the desired vBBU functionality (in this case the chaining is done automatically by OSM with the chaining rules already set for each VNF). These two options allow for SPs that do not have the expertise in telecommunications to use readily available vBBUs customized for particular types of applications or for SPs with the know-how to build their vBBU from scratch.

Each type of vBBU is specified as a Network Service (NS) descriptor, which are JSON data objects with standardized fields that OSM can interpret. The main fields in these describe the constituent VNFs of the vBBU and the connection between these VNFs. Similarly, the baseband VNF descriptors are JSON data objects. In this case, the main fields of these specify the computational resources, *i.e.*, network interfaces, storage, and processing, and the program(s) that the VNF must execute after its initialization. All descriptors are saved in an internal database in OSM and are available to SPs through the OSM client API.

OSM also performs the life-cycle management of vBBUs and vRRHs. In our prototype, this management is limited to the creation of vBBUs and vRRHs according to the available physical resources. The current release of OSM (version 4) lacks the capability to migrate baseband VNFs in response to events in the physical infrastructure that impact the vBBU performance. This functionality could be of great importance for commercial infrastructure providers that own multiple data centers and thousands of RRHs. However, we do not explore it in our prototype: it is within the scope of our future work.

The last functionality of OSM to be described here is its integration with OpenStack and ONOS. OSM has the instructions on how to build a vBBU based on the NS and VNF descriptors, which are used to control OpenStack to instantiate the baseband processing VNFs in the physical infrastructure. After the instantiation of all VNFs, OSM interacts with ONOS to configure the SDN network in such a way that the VNFs follow the chain of baseband processing of the vBBU.

4.2 RRH Hypervisor

The `RRH Hypervisor` in our implementation is based on our previous framework for RF front-end virtualization, named HYpervisor for software-Defined RAdio (HyDRA) [16], [27]. We have extended the last version with the extra functionalities required in a RANaaS system, with a northbound interface that exposes the slicing capabilities of HyDRA to OpenStack and the `vRAN Controller`. This extension enables the `vRAN Controller` to interact with HyDRA to create new vRRHs in their vRAN slices.

The architecture of HyDRA, extended with the northbound interface, is shown in Fig. 8. HyDRA adds a layer of indirection between the physical RRH and the vBBU executing in the CU/DU data centers. It is responsible for allocating spectrum resources to slices based on the request received from OpenStack, transforming them into virtualized resources, and assigning these virtual resources to a vRRH. This vRRH is then associated with a vBBU to

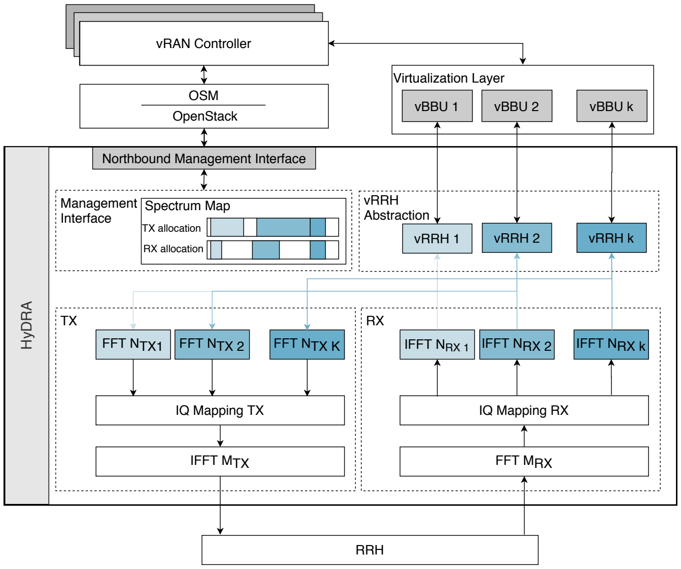


Fig. 8: Main architectural blocks of HyDRA.

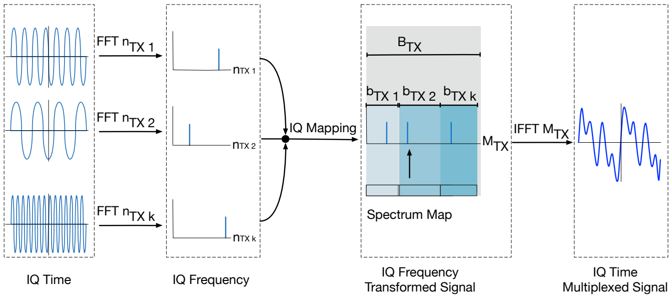


Fig. 9: Multiplexing process performed by the RRH Hypervisor.

realize a full-blown virtual base station. vBBUs operate as if they were interfacing directly with a standard RRH by sending/receiving IQ samples using the CPRI interface. HyDRA ensures isolation while allowing SPs to customize the vRRH transmission and reception parameters. Internally, HyDRA makes use of a transmission and a reception spectrum map to keep track of these configurations. It is important to highlight that the spectrum map is flexible in HyDRA. For example, a SP can request any center frequency or bandwidth, as long as the bands of operation of coexisting vRRHs do not overlap.

The core and challenging part in designing HyDRA was to multiplex the incoming IQ samples of each vRRH into a single signal that is transmitted by the physical RRH. Our multiplexing is based on FFT/Inverse FFT (IFFT) operations, as shown in Fig. 9. The frequency-domain decomposition and re-composition using the FFT and IFFT operations retain the transparency property of the virtualization. It ensures that the frequency components generated by vRRHs are always mapped to the same component in the physical spectrum of RRHs. Therefore, the signal received by mobile subscribers at the other end of the physical RRH would appear as if the SP owns a standard physical RRH (and similarly for the signal received by the physical RRH). In the remainder of this section, we will describe the operations performed by HyDRA, taking the downlink as an example, *i.e.*, from vRRHs to mobile subscribers. Also, we use uppercase symbols to denote parameters of the HyDRA and lowercase symbols each vRRH.

Initially, the incoming IQ samples from slice i are transformed from time to frequency domain in an FFT with $n_{\mathrm{TX\,i}}$ points. $n_{\mathrm{TX\,i}}$ is a function of the bandwidth of the vRRH and the sampling rate of the physical RRH. The resulting $n_{\mathrm{TX\,i}}$ frequency components are mapped into the buffer of an IFFT with M_{TX} points following the Spectrum Map configuration. Thus, $n_{\mathrm{TX\,i}}$ specifies the resolution used to multiplex the signal of the $i-th$ vRRH with the other slices coexisting in the physical RRH. HyDRA calculates $n_{\mathrm{TX\,i}}$ (Equation 1) as a rate of the bandwidth $B_{\mathrm{TX\,i}}$ with the total bandwidth B_{TX} and M_{TX}.

$$n_{\mathrm{TX\,i}} = \left\lceil \frac{B_{\mathrm{TX\,i}}}{B_{\mathrm{TX}}} \times M_{\mathrm{TX}} \right\rceil \quad \forall i \tag{1}$$

The value of the IFFT M_{TX} should be sufficiently large to cover the entire physical band used by the physical RRH with a frequency resolution that is equal or better than the frequency resolution of any instantiated vRRH. The value of M_{TX} is defined before during HyDRA's bootstrap and is manually defined. As a rule of thumb, its value is one of $[1024, 2048, 4098, 8196]$.

The mapping consists of moving the IQ samples of the FFT i to the correct bins of the IFFT M_{TX}. This process requires the definition the first and the last bin of the IFFT M_{TX} to which the $i-th$ vRRH is mapped, $m_{\mathrm{TX\,i,0}}$ and $m_{\mathrm{TX\,i,1}}$, respectively. The definition of these is a function of the center frequency F_{TX} and bandwidth B_{TX} of the RRH, and the center frequency $f_{\mathrm{TX\,i}}$ and bandwidth $b_{\mathrm{TX\,i}}$ of the $i-th$ vRRH, as shown in the Equation 2. After the mapping, we perform the IFFT to convert the frequency components of all vRRHs into the resulting multiplexed time-domain signal, which can be transmitted by the physical RRH.

$$\begin{aligned} m_{\mathrm{TX\,i,0}} &= \frac{f_{\mathrm{TX\,i}} - F_{\mathrm{TX}} - \frac{b_{\mathrm{TX\,i}}}{2} + \frac{B_{\mathrm{TX}}}{2}}{\frac{B_{\mathrm{TX}}}{M_{\mathrm{TX}}}} \\ m_{\mathrm{TX\,i,1}} &= m_{\mathrm{TX\,i,0}} + n_{\mathrm{TX\,i}} \end{aligned} \tag{2}$$

The multiplexing process based on FFT/IFFT operations is a perfect fit for our virtualization objectives. First, FFT/IFFT operations are low-level baseband processing operations agnostic of the access technology implemented by vBBU; this enables vBBUs to be connected with vRRH independently of the baseband processing chain. Second, modifications in the original signal caused by the multiplexing are not distinguishable from well-known wireless disturbances, *e.g.*, path-loss, frequency shift, and phase distortion; this allows receiving devices to recover the data transmitted using conventional physical layer equalization mechanisms. Third, FFT/IFFT are computationally efficient operations, highly optimized for modern processors through Single Instruction Multiple Data (SIMD); this allows the RRH Hypervisor to multiplex multiple vRRH simultaneously while using only a fraction of the resources of modern processors. However, using FFT/IFFT operations comes with some trade offs, of which the most impactful are: (*i*) the isolation between two vRRH is not ideal and the spectrum leakage from one vRRH can reduce the Signal-to-Interference-plus-Noise-Ratio (SINR) of other coexisting vRRH (we will explore this aspect in the following section, and (*ii*) the physical spectrum being abstracted must be contiguous and the total physical bandwidth is limited by the RF front-end and by the processing resources available to the RRH Hypervisor. Thus, two vRRHs using channels at different ends of the spectrum require more processing resources than two adjacent vRRHs.

4.3 Putting all together: creating a vRAN slice

So far, we have described each of the main components of AIRTIME's prototype. We now address how the main components of our prototype are integrated for the creation of a virtual base station within a vRAN slice. Starting with the request from the vRAN Controller, OSM must deploy all baseband processing VNFs of vBBU for the selected RAT and request the creation of a vRRH slice with HyDRA on top of the chosen physical RRH. After the successful deployment, ONOS automatically installs correct flows on the SDN switches so that all virtual elements are part of the same virtual network as the vRAN Controller. While the vRAN slice is active, the vRAN Controller can interact with the baseband processing VNFs and vRRHs to optimize any aspect of a virtual base station. The vRAN Controller can also interact with OSM to request the destruction of the virtual resources.

HyDRA can be turned into a VNF or be executed as standard software on top of the physical machine. As a rule of thumb, we want all HyDRA instances to be hosted in DUs data centers, *i.e.,* close to the physical RRH, while the baseband processing VNFs can be distributed between the same DU and CU data centers (there are of course some rare configurations to which this rule does not apply). Because HyDRA is responsible for the RRH slicing, its execution must outlive all baseband VNFs and should not be interrupted during the entire operation of AIRTIME. HyDRA can be installed as a GNURadio module or a standalone library or executable in any operating system that supports the C++ Standard Library. Its source code and the complete list of requirements are publicly available online in GitHub (https://github.com/maiconkist/gr-hydra). The complete source code is accompanied with several examples with configurable parameters such as the number of vRRHs, central frequency, and bandwidth of vRRH, and GUI elements such as spectrum waterfall and plotters to visualize data transmitted and received. In the next section, we present the experimental evaluation of AIRTIME's prototype.

5 Experimental Proof of Concept

In this section, we show the evaluation of the prototype described in the previous section. Our goal is to validate the RRH and BBU virtualization capabilities of AIRTIME, regarding performance, isolation, and scalability. Furthermore, we illustrate how AIRTIME address the three key challenges in RANaaS: slice isolation, slice programmability, and slice adaptability.

5.1 Experimental Setup

The experimental setup is shown in Fig. 10. Acting as CU and DU data centers, we have two laptops with an Intel i5-6440HQ processor, 8GBs of memory, using the operational system Ubuntu 18.04 with the latest updates installed. These laptops are connected to a Dell SDN switch model S4048T-ON, which acts as the fronthaul network. An Ettus Universal Software Radio Peripheral (USRP) model B210 operates as a RRH. This USRP can tune into any frequency from 70 MHz to 6000 MHz, with a maximum instantaneous bandwidth of 56 MHz. In practice, the maximum instantaneous bandwidth represents the bandwidth available to HyDRA when it creates vRRHs. More precisely, the bandwidth of all vRRHs multiplexed in the same USRP cannot exceed this value, and all vRRHs' channels must be located inside the instantaneous bandwidth range.

We have installed LXC in CU and DU data centers, so both can run the baseband processing VNFs. We selected the laptop acting as CU to install OSM and OpenStack, so it can work as the BBU Hypervisor, managing all baseband processing VNFs in the CU and in the DU data center. Two vBBU NSs are installed in OSM, representing two services aligned with the capabilities of each air-interface. The first one is an LTE vBBU that communicates with mobile subscribers. This vBBU is built with the aggregation of three baseband processing VNFs, namely low-PHY VNF, high-PHY VNF, and MAC VNF. The second is a NarrowBand-IoT vBBU that comprises only one baseband processing VNF, which implements the entire RAT to communicate with IoT sensors [35]. The RRH Hypervisor, *i.e.,* HyDRA, is installed only in the DU data center. We emphasize that our experimental setup is restricted to the RAN, thus not comprising the CN of the SPs. In this case, we do not have a full stack for LTE and NarrowBand-IoT because we aim to validate the RRH and BBU virtualization capabilities of AIRTIME. Moreover, the CU and DU are separated from each other by 10 meters and connected through a 10 Gbps Ethernet link with an average network latency of 4 ms. Table 1 summarizes the main radio parameters for the LTE and NarrowBand-IoT vBBUs, as well as for HyDRA.

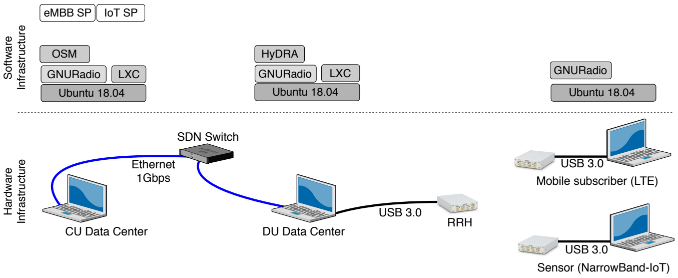


Fig. 10: Experimental setup.

Finally, we have two mobile users for LTE and NarrowBand-IoT RATs. Each user comprises an Ettus USRP B200 and a laptop running Ubuntu 18.04 with the latest updates and same hardware configuration as the DU. The distance between the RRH and the user's antenna is exactly 5 meters. The users do not run the BBU and RRH Hypervisors; our solution is designed to work with legacy

TABLE 1: eMBB SP, IoT SP, and HyDRA configurations

SPs	vBBU	VNFs	Parameters
eMBB	LTE (GNURadio based)	VNF 1: Low-PHY VNF 2: High-PHY VNF 3: MAC	**CF**: 947 MHz, **BW**: 1.4 MHz, **FFT**:128, **CP**: 7 symbols (short), **MOD**:QPSK
IoT	NarrowBand-IoT (GNURadio based)	VNF 1: IoT (PHY & MAC)	**CF**: 951MHz, **BW**: 200 KHz, **FFT**:64, **CP**: 7 symbols (short), **MOD**:BPSK
–	–	HyDRA	**CF TX/RX**: 950 MHz, **BW TX/RX**: 8 MHz, **FFT M TX/RX**: 4096

end-user devices that need not be aware of the virtualisation occurring in the network.

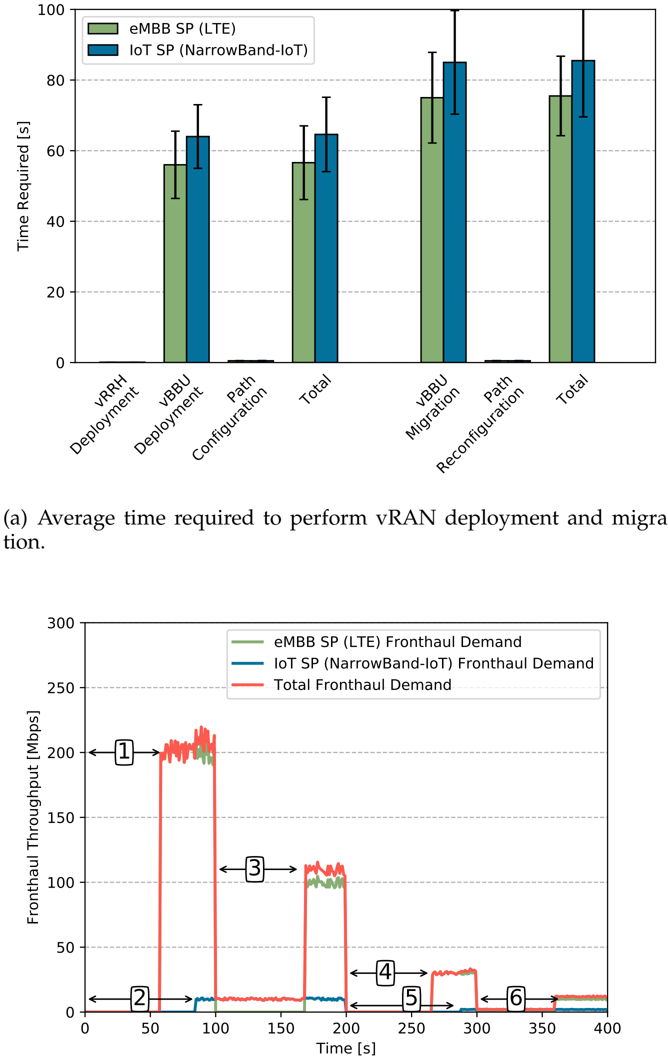
We envisioned a scenario whereby an "eMBBC SP" and an "IoT SP" interface with OSM to request the creation of slices based on their particular application requirements. The `vRAN Controllers` of each SP interacts with the OSM to perform the following changes:

- Step ① (start at t = 0s): eMBBC SP requests the creation of a vRAN with an LTE coverage. All three baseband processing VNFs of the LTE vBBU are instantiated in the CU data center.
- Step ② (start at t = 0s): The IoT SP requests the creation of a vRAN with a NarrowBand-IoT coverage. The baseband processing VNF for this RAT is instantiated in CU.
- Step ③ (start at t = 100s): The low-PHY baseband processing VNF of the LTE vBBU is moved from CU to DU.
- Step ④ (start at t = 200s): The high-PHY baseband processing VNF of the LTE vBBU is moved from CU to DU.
- Step ⑤ (start at t = 200s): The NarrowBand-IoT baseband processing VNF is moved from CU to DU.
- Step ⑥ (start at t = 300s): The MAC VNF of the LTE vBBU is moved from CU to DU.

This experimental setup also shows that AIRTIME addresses two key challenges of RANaaS: (*i*) programmability, by having SPs with vRANs tailored to their particular service requirements, and (*ii*) adaptability, by having the baseband processing VNFs being migrated to increase resource usage efficiency. However, it is expected that in a real-life scenario, the `vRAN Controller` acts based on events occurring in the vRAN, instead of working with the pre-determined timestamps of our experimental setup.

5.2 Performance Analysis

We split our performance analysis into two categories: (*i*) infrastructure level (subsection 5.2.1), and (*ii*) service level (subsection 5.2.2). At the infrastructure level, we are interested in obtaining insights about two essential metrics: the time required to create a vRAN considering vRRH and vBBU deployments as well as path configuration; and the fronthaul throughput required between CU and DU for different baseband processing VNF distributions for this vRAN. At the service level, we analyze more two important end-to-end metrics: the throughput and the latency experienced by the end-users of the LTE and IoT SPs.



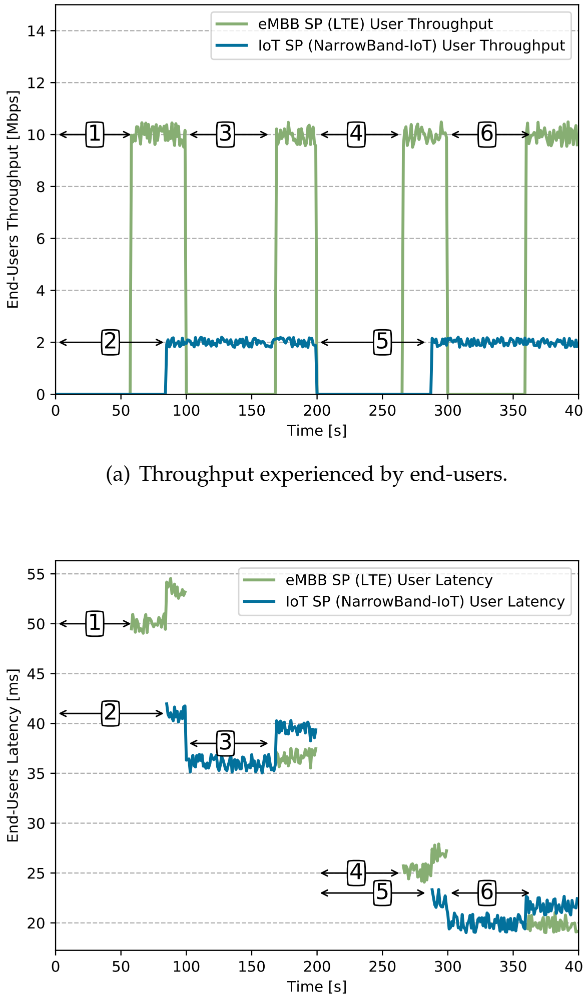
(a) Average time required to perform vRAN deployment and migration.

(b) Fronthaul throughput required for different VNF distributions.

Fig. 11: Infrastructure level evaluation of AIRTIME.

5.2.1 Infrastructure Level Performance

The deployment of the LTE and NarrowBand-IoT vRANs start at time t = 0s. Fig. 11(a) provides insights regarding the time taken, considering the three main operations performed: vRRH deployment, vBBU deployment, and path configuration. These results are shown with a 95% confidence interval. We can see that deploying vBBUs is a time demanding operation, taking up to 60 s for LTE and 85 s for NarrowBand-IoT. The vRRH deployment and path



(a) Throughput experienced by end-users.

(b) Latency experienced by end-users.

Fig. 12: Service level evaluation of AIRTIME

configuration operations take less than 1 s, thus accounting for only a fraction of the total time required to deploy the vRANs. In the same figure, we also show the average time needed for migrating the vBBUs, together with the time required for the path reconfiguration. We can see that vBBU migration takes a longer time than its initial deployment. This behavior occurs because the migration requires the additional transfer of the baseband processing VNF state, *i.e.,* processes running, files opened, and network connections.

The throughput required for each slice, between the CU/DU, is shown in Fig. 11(b) (these values were obtained in one of our experiment executions to illustrate the experiment behavior). The circled numbers indicate when the steps described in the experimental setup are completed. From time t = 0s until step ① completes, we can see that no traffic goes through the fronthaul, followed by a spike in the traffic when the LTE IQ traffic starts (approximately 200 Mbps). When both vBBUs are running in the CU (after the completion of step ②), the IQ transfer requires approximately 230 Mbps. The migration of the low-PHY LTE VNF starts at t = 100s (step ③), which reflects in the LTE IQ traffic stopping until it is completed at t = 160s. The same behavior can be observed in the subsequent migration steps for the LTE vBBU (steps ④ and ⑥) and for the NarrowBand-IoT vBBU (steps ② and ⑤). From this evaluation, we can observe that offloading the baseband processing VNFs to the DU supports the adoption of fronthaul links with low bandwidth capacity.

5.2.2 *Service Level Performance*

Next, we look at the throughput and latency experienced by the end-users, as shown in Fig. 12 (these values were obtained from the same experiment execution mentioned previously). The throughput, shown in Fig. 12(a), is almost constant for both of the end-users for all baseband processing VNF distributions. Let us take the LTE mobile subscriber as an example: the throughput is constant for all baseband processing VNFs distribution options, independent of the level of centralization (after completion of steps ①, ③, ④, and ⑥). The same can be observed for the NarrowBand-IoT sensor (after completion of steps ② and ⑤). This behavior is because the bottleneck is not the fronthaul link but rather the LTE and NarrowBand-IoT RATs, which with the configuration we used can achieve only 10 Mbps and 2 Mbps, respectively.

The latency experienced by the end-users is shown in Fig. 12(b). Migrating the baseband processing VNFs to the DU significantly reduces the latency. In this case, let us take the IoT sensor as an example: we have a latency of approximately 42 ms when the NarrowBand-IoT baseband processing VNF is running in the CU together with all LTE VNFs (after step ② completes). This latency drops to 36 ms when the LTE vBBU is being migrated (at t = 100 s). It then increases again when the LTE vBBU restarts in step ③, but now with the low-PHY VNF executing in the DU, thus forwarding less data through the fronthaul link. A similar behavior can be noticed when the NarrowBand-IoT VNF is running in the DU (after step ⑤ completion). This evaluation also shows another benefit of offloading the baseband processing VNFs to the DU: it can enable an URLLC slice due to the proximity to the end-users. However, the caveat when the baseband processing VNFs are moved from the CU to the DU is the loss of advanced Large-Scale Cooperative Multiple Antenna Processing (LS-CMA) mechanisms, which we do not explore in AIRTIME's prototype due to their high implementation complexity.

So far, we have evaluated the performance of AIRTIME's prototype, considering only high-level metrics for infrastructure and SPs. The results so far show that AIRTIME enables multiple heterogeneous vRAN slices on top of the physical infrastructure. Next, we focus our analysis evaluating the main novelty of AIRTIME: the `RRH Hypervisor` and its capabilities to create multiple vRRHs.

5.3 Isolation

In this analysis, we are interested in addressing the remaining challenge in RANaaS: slice isolation, *i.e.,* the isolation between vRRHs. As HyDRA multiplexes vRRHs in the frequency domain, the best way to measure their isolation is considering different frequency separations among them, *i.e.,* guard-bands. More precisely, we measure the SINR at the mobile subscriber and the NarrowBand-IoT sensor receiver, for different guard-bands between the vRRHs bound to the LTE and NarrowBand-IoT vBBUs.

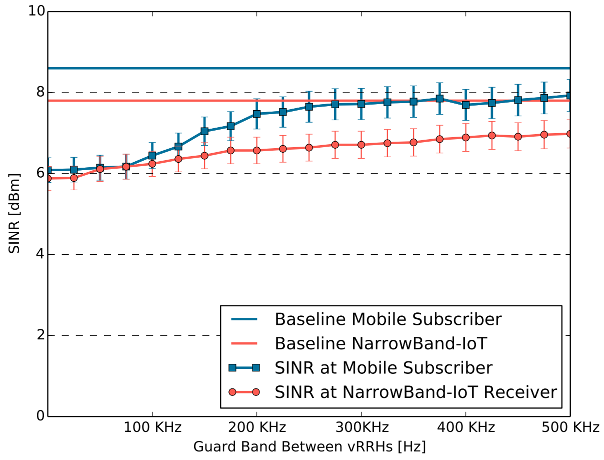


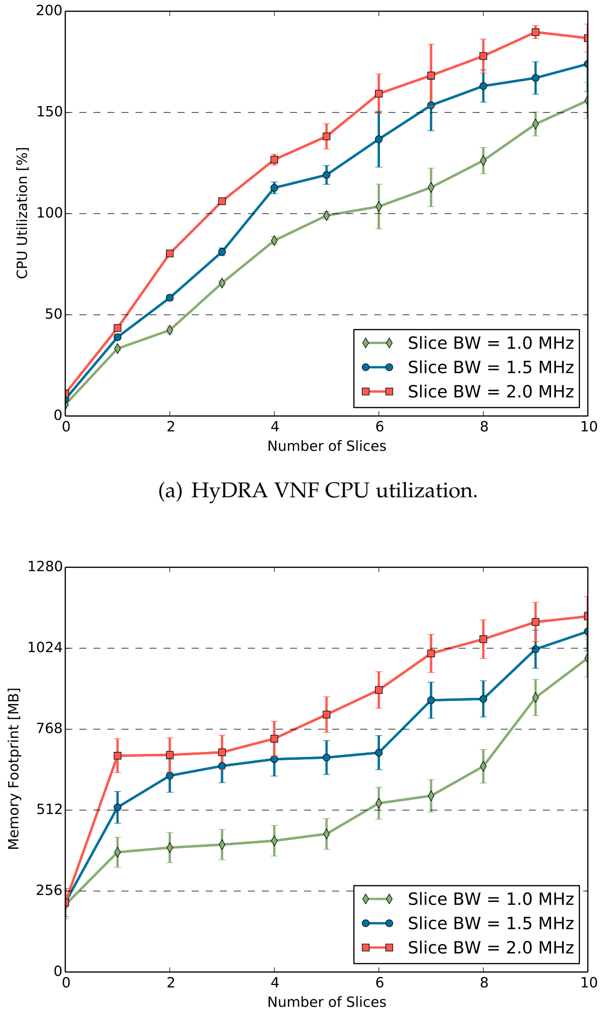
Fig. 13: SINR observed at the receiver as a function of the guard-band between the vRRHs.

The results obtained are shown in Fig. 13. The lines labeled *Baseline Mobile Subscriber* and *Baseline NarrowBand-IoT* show the SINR at clients when using a physical RRH without the `RRH Hypervisor`. We compare against the case where HyDRA is used to support LTE and NarrowBand-IoT vBBUs simultaneously. When HyDRA is used, we can see that guard-bands smaller than 200 KHz cause a reduction in SINR up to 3 dBm. This occurs due to the combination of spectrum leakage from the USRP and from the FFT/IFFT operations performed in the hypervisor. The interference due to the USRP leakage diminishes for guard-bands greater than 200 KHz; however, the leakage from FFT/IFFT continues for larger guard-bands, reducing the SINR by up to 2 dBm. These results show that HyDRA can create isolated vRRHs, but with some limitations that arise from the RF front-end of the RRH and HyDRA's implementation. Moreover, we show that HyDRA ensures isolation by multiplexing vRRHs in the frequency domain, while allowing the configuration of central frequency, bandwidth, and transmission/reception gain on-the-fly.

5.4 Scalability

In our third analysis, we quantify how AIRTIME scales in terms of processing and memory requirements, *i.e.*, the increase of resources used as vRANs are deployed. Arguably, the bottleneck in AIRTIME is the `RRH Hypervisor` VNF instance because: (*i*) only one instance is bound to a physical RRH. Thus, this single VNF must cope with the processing demands of multiple vRRHs, and (*ii*) this instance must multiplex the IQ samples from multiple vRRHs using only the processing resources of one physical machine (different from vBBUs, which can be distributed among multiple data centers and physical machines). For these reasons, we focus only on the analysis of the resources used by HyDRA. Moreover, there are two main parameters that impact the performance of HyDRA and that we will explore: (*i*) the number of vRRHs and (*ii*) the bandwidth used by each vRRH.

To assess the overhead incurred for the virtualization operations performed by HyDRA, we used a setup where we start from zero vRANs and request the creation of one vRAN up to a total of ten vRANs, all using the same bandwidth [1 MHz, 1.5 MHz, and 2 MHz]. We saturate the traffic of each slice with TCP traffic (similar to the previous setup) and measure the CPU usage (using the `mpstat` tool) and memory footprint (using the `vmstat` tool) of the VNF running HyDRA.



(a) HyDRA VNF CPU utilization.

(b) HyDRA VNF memory footprint.

Fig. 14: CPU and memory footprint of the HyDRA VNF for varying number of slices and slice bandwidths.

In Fig. 14(a) we show the results for the CPU usage as a percentage of one core in the Intel i5-6440HQ processor, whereas Fig. 14(b) shows the total memory used in MB. Apart from the initial overhead associated with the creation of the first vRRH, the creation of extra vRRHs incurs in small and almost constant increments in both CPU and memory. The initial increase is due to HyDRA internal operations that are inactive when there are no vRRHs (construction of the spectrum map of vRRHs, FFT, IFFT, and IQ mapping).

The almost constant rise in CPU utilization after the first vRRH is a result of the additional FFT and internal buffers required for each new vRRH. As we deploy vRANs, the CPU utilization increases to the point that the baseband processing VNFs cannot share the physical machine of HyDRA without scaling issues. For example, in the commodity machine used in our experiment, if we deploy 5 vRANs with 1 MHz each, the hypervisor requires up to one full core of the processor, leaving only the other core for the baseband processing of the vBBUs. The actual number of slices that can be supported depends both on the resources available for HyDRA and the capabilities of the physical RRH.

To conclude, we have shown that our prototype can support the deployment of multiple heterogeneous vRANs, satisfying the main requisites for a RANaaS system: slice isolation, slice programmability, and slice adaptability. In the following section, we survey research endeavors to enable RANaaS on future mobile networks. These endeavors have exposed the virtualization limitations of current approaches.

5.5 Qualitative Benefits

We summarize the main qualitative benefits of the virtualization design adopted in AIRTIME.

- **RRH virtualization:** AIRTIME enables the virtualization of RRH using the novel `RRH Hypervisor`. Our approach for RRH virtualization is state-of-the-art. RRH virtualization is a significant step towards future multi-tenancy networks, as being introduced by 3GPP [36].
- **Multi-radio access networks:** AIRTIME simplifies the integration and operation of multi-RATs on top of one physical RAN infrastructure. In current centralized baseband architectures, an RRH is mapped to only one (v)BBU and can only operate one access technology. AIRTIME enables RRHs to connect to multiple vBBUs and operate in any technology merely by creating a new vRRH.
- **Flexible multi-tenant access networks:** Current multi-tenant access networks are restricted by the underlying RAT of the physical RAN. In contrast, AIRTIME enables multiple SPs to run a full-blown vRAN tailored to their service requirements on top of the physical RAN.
- **CAPEX reduction:** AIRTIME can run on top of the infrastructure of centralized baseband architectures while at the same time enabling multiple vRANs to share the same RAN. Consequently, SPs can deploy tailored RATs only by creating a new vRRH-vBBU pair and multiplex it through HyDRA.

6 Related Work

The state-of-the-art on RAN slicing can be traced back to the initial efforts on active RAN sharing [36], [39]. Two approaches, named Multi-Operator Core Network (MOCN) and Multi-Operator Radio Access Network (MORAN), were considered back then. More attention has been given to the MOCN approach [8], [10], which has been standardized in LTE Release 8. While both approaches consider the slicing and virtualization of CN, MORAN extends the slicing capabilities to RAN with a dedicated radio channel for each slice owner [5], [6], [7], [9], [26], [37]. The primary focus of these works is on the design of efficient radio resource scheduling algorithms across multiple virtual network slices. The term "virtualization" in some of these works is used loosely to refer to the performance isolation perceived by mobile subscribers of each slice sharing the RAN resources [40].

Table 2 summarizes some of the most notable research efforts in RAN virtualization considering six aspects:

- **BBU virtualization:** this refers to the ability to execute, at least, the softwarized version of BBUs on top of standard data center hardware. This is a fundamental aspect of any RANaaS system.
- **Functional split:** refers to the ability to distribute vBBU functions between CU and DU data centers. AIRTIME is the only RANaaS system that considers this aspect in vBBUs.
- **RRH virtualization:** refers to the capability to slice the physical RRH into vRRHs (we do not consider research that virtualizes at the MAC layer). AIRTIME is, to the best of our knowledge, the only RANaaS system that enables RRH virtualization.
- **Multiple technologies:** is the capability to run multiple RATs on top of the physical RRHs. Again, AIRTIME is the only system that enables this aspect.
- **Radio resource:** refers to the resource type that is abstracted and used by the slice owners. Some systems chose to allocate PRBs, while others allocate spectrum directly. The latter is arguably more flexible.
- **Slice programmability:** refers to the lowest layers of the RAT that the slice owners can customize. A RANaaS system should enable customization from the low-PHY layer so that slice owners can tailor the RAT to any service requirement.
- **Evaluation:** refers to the evaluation methodology, *e.g.*, simulation or experimental, of the proposed solution. Most of the research efforts present some form of experimental evaluation.

A system that supports RAN virtualization through resource sharing is Orion [26], [37] (and its predecessor FlexRAN [9]). These systems have a clear separation of the control and data planes of base stations. For the control plane, Orion introduces a master controller that interacts with high-level applications, and for the data plane, it presents a southbound API between the master controller and all base stations. Network slicing is achieved by enabling high-level applications to define the way the radio resources need to be allocated among the mobile subscribers based on application requirements. The master controller is responsible for performing the corresponding control operations. Orion constrains all slice owners to the same RAT of the underlying base stations. Furthermore, the control plane capabilities are limited by the APIs presented by the master controller. In contrast, AIRTIME enables slice programmability so that slice owners can tailor any aspect of their vRAN according to their needs.

Other RAN slicing approaches seek full isolation by running the virtual base stations instances of a slice as VNFs [6], [7] moving to fill the need for slice programmability. These

TABLE 2: Overview of related work in the field of RAN virtualization.

Proposal	BBU virtualization	Functional split	RRH virtualization	Multiple technologies	Radio resource	Slice programmability	Evaluation
NVS [5]	✓				PRB	MAC layer	Experimental
FlexRAN [9]	✓				PRB	MAC Layer	Experimental
Orion [26] [37]	✓				PRB	MAC Layer	Experimental
SoftAir [7]	✓				Spectrum	PHY/MAC	NA
Concert [6]	✓				Spectrum	PHY/MAC	NA
Mendes *et al.* [38]	✓		✓	✓	Spectrum	PHY	Experimental
AIRTIME	✓	✓	✓	✓	Spectrum	PHY/MAC	Experimental

works assume a dedicated RRH for each slice, which has the downside of inefficient use of radio resources. Nevertheless, wireless virtualization surveys and position papers advocate for this functional isolation [21]. In AIRTIME, we go further in this approach and present an innovative RRH virtualization layer to allow multiple slices to share the physical RRH without compromising their programmability. To the best of our knowledge, the research reported in this article is the only work that has addressed RAN slicing with full programmability to slice owners, through a design perspective and prototype implementation.

Finally, the proposal of Mendes *et al.* [38] shares the fundamental concepts of BBU and RRH virtualization with AIRTIME. However, due to the lack of the `RRH Hypervisor`, this proposal can provide only one RAT on top of each RRH. Arguably, this significantly hinders the programmability and adaptability of vRAN, as each physical RRH can be assigned to only one vRAN, thus hindering the programmability and adaptability of other coexisting SPs. Moreover, Mendes *et al.* [38] did not explore virtual machine or container placement and split options in their research options. Instead, their focus is in sharing a common RF front-end (as done in HyDRA, but with another multiplexing method). Moreover, their proposal does not implement any of the interfaces that allow SP to manage physical RANs and SPs to manage the vRAN slices and BBU virtualization is limited to the PHY layer *i.e.,* all baseband processing is centralized and the functional split is not possible. In contrast, AIRTIME leverages BBU and RRH virtualization to realize a multi-service RANaaS system that allows SPs to control access and manage their vRAN slice.

In summary, the solutions found in the literature present the following limitations: (*i*) restrict all slice owners to the same RAT of the underlying base stations and the control plane capabilities are limited by APIs presented by a master controller [26], [37], (*ii*) assume a dedicated RRH for each slice, leaving the inefficient use of radio resources [6], [7], (*iii*) provide only one RAT on top of each RRH [38]. Considering these limitations of the state-of-the-art, AIRTIME achieves empowering slice programmability so that slice owners can tailor any aspect of their vRAN according to their requirements. Moreover, AIRTIME offers an innovative RRH virtualization layer to provide multiple slices to share the physical RRH without compromising their programmability. Furthermore, AIRTIME leverages BBU and RRH virtualization to realize a multi-service RANaaS system that will enable SPs to control access and manage their vRAN slice.

7 Concluding Remarks

In this article, we presented AIRTIME, a RAN slicing system for future mobile networks. Heterogeneous virtual RANs can be deployed dynamically over the network infrastructure, co-existing in a fully isolated manner in terms of networking, processing, and radio resources. At the same time, AIRTIME enables slice owners to customize any aspect of their virtual RAN, from the core network layout to the RAT provided to end-users. We developed a prototype implementation to show how AIRTIME can enable the provisioning of multiple services with tailored RANs on top of a single physical RAN infrastructure. We assessed the benefits and impact of our prototype by demonstrating that: (*i*) multiple heterogeneous RAN slices can be dynamically instantiated on top of the physical infrastructure in a couple of seconds, (*ii*) RAN slices can be tailored for particular service requirements by moving baseband processing functions close to the end-users, and (*iii*) RAN slices are isolated from each other at all levels of the infrastructure. All these capabilities together are a step towards realizing the RAN-as-a-Service paradigm.

We expect that AIRTIME can catalyze mobile network innovations in a range of areas, from the introduction of new air-interfaces specialized in specific services, to management of data center and fronthaul resources. RAN slicing solves the challenges of current centralized baseband architectures while enabling unprecedented control over any aspect of the physical infrastructure. Moreover, researches using AIRTIME should investigate security and reliability since AIRTIME can avoid and prevent common cyberattacks, such as authentication and access control, for example.

Acknowledgements

This work has received funding from the European Union's Horizon 2020 Program Framework for research and technological development under the grant agreements No. 732497 (5GInFire) and No. 732174 (ORCA). It has also been supported by the Science Foundation Ireland under grant 13/RC/2077, SFI Research Centre CONNECT.

References

[1] C. Sexton *et al.*, "5G: Adaptable Networks Enabled by Versatile Radio Access Technologies," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 688–720, 2017.

[2] ITU, "IMT Vision – Framework and overall objectives of the future development of IMT for 2020 and beyond," 2015.

[3] I. Quintana-Ramirez *et al.*, "The Making of 5G: Building an End-to-End 5G-Enabled System," *IEEE Communications Standards Magazine*, vol. 2, no. 4, pp. 88–96, 2019.

[4] O. Adamuz-Hinojosa *et al.*, "Harmonizing 3GPP and NFV Description Models: Providing Customized RAN Slices in 5G Networks," *IEEE Vehicular Technology Magazine*, vol. 14, no. 4, pp. 64–75, Dec 2019.

[5] R. Kokku *et al.*, "NVS: A Substrate for Virtualizing Wireless Resources in Cellular Networks," *IEEE/ACM Transactions on Networking*, vol. 20, no. 5, pp. 1333–1346, 2012.

[6] J. Liu *et al.*, "CONCERT: A Cloud-Based Architecture For Next-Generation Cellular Systems," *IEEE Wireless Communications*, vol. 21, no. 6, pp. 14–22, 2014.

[7] I. F. Akyildiz, P. Wang, and S.-C. Lin, "SoftAir: A Software Defined Networking Architecture for 5G Wireless Systems," *Computer Networks*, vol. 85, no. 7, pp. 1–18, 2015.

[8] T. Taleb *et al.*, "EASE: EPC As a Service to Ease Mobile Core Network Deployment Over Cloud," *IEEE Network*, vol. 29, no. 2, pp. 78–88, 2015.

[9] X. Foukas *et al.*, "FlexRAN: A Flexible and Programmable Platform for Software Defined Radio Access Networks," in *Emerging Networking EXperiments and Technologies*, 2016, pp. 427–441.

[10] P. Rost *et al.*, "Network Slicing to Enable Scalability and Flexibility in 5G Mobile Networks," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 72–79, 2017.

[11] Z. Zaidi, V. Friderikos, and M. A. Imran, "Future RAN Architecture: SD-RAN Through a General-Purpose Processing Platform," *IEEE Vehicular Technology Magazine*, vol. 10, no. 1, pp. 52–60, March 2015.

[12] A. Papa *et al.*, "Optimizing Dynamic RAN Slicing in Programmable 5G Networks," in *IEEE International Conference on Communications (ICC)*, May 2019, pp. 1–7.

[13] X. Jin *et al.*, "SoftCell: Scalable and Flexible Cellular Core Network Architecture," in *Conference on Emerging Networking Experiments and Technologies*, 2013, pp. 163–174.

[14] K. Pentikousis, Y. Wang, and W. Hu, "MobileFlow: Toward Software-Defined Mobile Networks," *IEEE Communications Magazine*, vol. 51, no. 7, pp. 44–53, 2013.

[15] 3GPP, "Architecture Enhancements for Dedicated Networks. TS 23.707," 3rd Generation Partnership Project, Tech. Rep., 2017.

[16] M. Kist *et al.*, "SDR Virtualization in Future Mobile Networks: Enabling Multi-Programmable Air-Interfaces," in *IEEE International Conference on Communications*, 2019, pp. 1–6.

[17] J. Santos *et al.*, "Towards Enabling RAN as a Service - The Extensible Virtualisation Layer," *IEEE International Conference on Communications (ICC)*, 2019.

[18] M. A. Marotta *et al.*, "Characterizing the Relation Between Processing Power and Distance Between BBU and RRH in a Cloud RAN," *IEEE Wireless Communications Letters*, vol. 7, no. 3, pp. 472–475, 2018.

[19] M. Kist *et al.*, "Flexible Fine-Grained Baseband Processing with Network Functions Virtualization: Benefits and Impacts," *Elsevier Computer Networks*, vol. 151, pp. 158–165, 2019.

[20] A. S. Thyagaturu, Z. Alharbi, and M. Reisslein, "R-FFT: Function Split at IFFT/FFT in Unified LTE CRAN and Cable Access Network," *IEEE Transactions on Broadcasting*, vol. 64, no. 3, pp. 648–665, Sep. 2018.

[21] A. Checko *et al.*, "Cloud RAN for Mobile Networks: A Technology Overview," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 405–426, 2015.

[22] C. Liang and R. Yu, "Wireless Network Virtualization: A Survey, Some Research Issues and Challenges," *IEEE Communications Surveys & Tutorials*, vol. 17, 2014.

[23] X. Foukas *et al.*, "Network Slicing in 5G: Survey and Challenges," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 94–100, 2017.

[24] 3GPP, "Summary of RAN3 status on CUDU split option 2 and option 3, and questions/issues for RAN2," 3rd Generation Partnership Project, Tech. Rep., 2017.

[25] L. M. 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.

[26] X. Foukas, M. K. Marina, and K. Kontovasilis, "Orion: RAN Slicing for a Flexible and Cost-Effective Multi-Service Mobile Network Architecture," in *International Conference on Mobile Computing and Networking*, 2017, pp. 127–140.

[27] M. Kist *et al.*, "HyDRA: A hypervisor for software defined radios to enable radio virtualization in mobile networks," in *IEEE Conference on Computer Communications Workshops*, 2017, pp. 960–961.

[28] S. Perez, J. M. Cabero, and E. Miguel, "Virtualization of the Wireless Medium: A Simulation-Based Study," in *IEEE Vehicular Technology Conference*, 2009, pp. 1–5.

[29] K. Nakauchi, Y. Shoji, and N. Nishinaga, "Airtime-based resource control in wireless LANs for wireless network virtualization," in *International Conference on Ubiquitous and Future Networks*, 2012, pp. 166–169.

[30] X. Lu, K. Yang, and H. Zhang, "An Elastic Sub-carrier and Power Allocation Algorithm Enabling Wireless Network Virtualization," *Wireless Personal Communications*, vol. 75, no. 4, pp. 1827–1849, 2014.

[31] J. Sachs and S. Baucke, "Virtual radio: a framework for configurable radio networks," *International Conference on Wireless Internet*, pp. 1–7, 2008.

[32] Y. Zaki, *Future Mobile Communications: LTE Optimization and Mobile Network Virtualization*. Springer Science & Business Media, 2012.

[33] A. Gudipati *et al.*, "SoftRAN: Software Defined Radio Access Network," in *ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking*. New York, USA: ACM Press, 2013, p. 25.

[34] O. Font-Bach *et al.*, "When SDR meets a 5G candidate waveform: Agile use of fragmented spectrum and interference protection in PMR networks," *IEEE Wireless Communications*, vol. 22, no. 6, pp. 56–66, 2015.

[35] P. Andres-Maldonado *et al.*, "Narrowband IoT Data Transmission Procedures for Massive Machine-Type Communications," *IEEE Network*, vol. 31, no. 6, pp. 8–15, November 2017.

[36] K. Samdanis, X. Costa-Perez, and V. Sciancalepore, "From network sharing to multi-tenancy: The 5G network slice broker," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 32–39, 2016.

[37] X. Foukas *et al.*, "Experience Building a Prototype 5G Testbed," in *Proceedings of the Workshop on Experimentation and Measurements in 5G*. ACM, 2018, pp. 13–18.

[38] J. Mendes *et al.*, "Cellular Access Multi-Tenancy Through Small Cell Virtualization and Common RF Front-End Sharing," *Computer Communications*, vol. 133, pp. 59–66, January 2019.

[39] X. Costa-Perez *et al.*, "Radio access network virtualization for future mobile carrier networks," *IEEE Communications Magazine*, vol. 51, no. 7, pp. 27–35, 2013.

[40] J. van de Belt, H. Ahmadi, and L. E. Doyle, "Defining and Surveying Wireless Link Virtualization and Wireless Network Virtualization," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1603–1627, 2017.

Maicon Kist is a Ph.D student in Computer Science at the Institute of Informatics (INF) of the Federal University of Rio Grande do Sul (UFRGS), Brazil. He received his M.Sc. title in from INF of UFRGS (2012), Brazil. In addition, he holds a B.Sc. in Computer Science from the University of Santa Cruz do Sul (2008), Brazil. His research involves Wireless Networks, Next Generation Networks, Software Defined Radio, and Cognitive Radio Networks.

Joao F. Santos is pursuing a Ph.D on wireless networks at the CONNECT Telecommunications Research Centre, headquartered at Trinity College Dublin. He holds a B.Sc. in Telecommunications Engineering from Universidade Federal Fluminense (2016). He worked at Rede Nacional de Ensino e Pesquisa (RNP) as the main developer of the Clearinghouse of the FIBRE testbed federation. His research interests include network slicing, radio virtualisation, and end-to-end network orchestration.

Diarmuid Collins received a Ph.D. degree in computer science in 2016 from Trinity College Dublin, Ireland. He received a B.Sc. from Waterford Institute of Technology in 2003, and an MSc degree in Software Engineering and Database Systems from NUI Galway in 2006. In 2016, he joined the CONNECT Telecommunications Research Centre where he has been working broadly on wireless networks, network control and resource management, remote experimentation, eLearning, 5G, NFV, and ICN.

Juergen Rochol is an associate professor at the Institute of Informatics of the Federal University of Rio Grande do Sul (UFRGS), Brazil. He received his M.Sc. degree in Physics, and his Ph.D. degree in Computer Science, both from UFRGS in 1972 and 2001, respectively. His research interests include wireless networks, next generation networks, optical networks and traffic control on broadband computer networks.

Luiz A. DaSilva holds the chair of Telecommunications at Trinity College Dublin. His research focuses on distributed and adaptive resource management in wireless networks, and in particular wireless resource sharing, dynamic spectrum access, and the application of game theory to wireless networks. He is the Director of CONNECT, the Science Foundation Ireland Research Centre for Future Networks and Communications, a Fellow of Trinity College Dublin, and an IEEE Fellow.

Cristiano Bonato Both is an associate 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 network, and SDN-like solutions for the Internet of Things. He is participating in several Technical Programme and Organizing Committees for different worldwide conferences and congresses.