Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

PHD THESIS

Virtual Radios, Real Services: Enabling End-to-End Network Slicing through Radio Virtualisation

Author:
Joao Felipe Faco Cals Cruz SANTOS

Supervisor:
Prof. Anil KOKARAM

Co-Supervisor:
Prof. Luiz A. DASILVA

August 24, 2021

Declaration

I declare that this thesis has not been submitted as an exercise for a degree at this or any other university and is entirely my own work.

I agree to deposit this thesis in the University's open access institutional repository or allow the Library to do so on my behalf, subject to Irish Copyright Legislation and Trinity College Library conditions of use and acknowledgement.

I consent to the examiner retaining a copy of the thesis beyond the examining period, should they so wish (EU GDPR May 2018).

Signed:

Joao F. Santos, August 24, 2021

> “*I may not have gone where I intended to go,*
>
> *but I think I have ended up where I needed to be.*”
>
> Dirk Gently, The Long Dark Tea-Time of the Soul, *1988*

Summary

This work aims to resolve some resource management problems that arise due to changes in the mobile network market and business models, brought by recent trends in commercial mobile networks, such as the widening range of services offered by Mobile Network Operators (MNOs), and the increasing importance of vertical industries and service providers as sources of revenue. These changes led the 3GPP to introduce network slicing, proposing a logical division of the mobile network infrastructure into independent virtual networks, known as Network Slices (NSs). Each NS acts as a completely isolated End-to-End (E2E) network, tailored with different resources and functionality to support applications with specific Quality of Service (QoS) requirements. Similarly to the separation of mobile networks into Core Network (CN) and Radio Access Network (RAN) segments, the offer of NSs can be split into CN as a Service (CNaaS) and RAN as a Service (RANaaS), allowing MNOs to provide their clients with custom CN slices, custom RAN slices, or a combination of both, forming fully custom E2E NSs. In particular, this thesis studies the resource management functionality for enabling E2E network slicing through radio virtualisation. The problems we address in this thesis can be summarised in the form of the following research questions:

- What are the requirements for using radio hypervisors to support RANaaS?
- How can radio hypervisors provision and instantiate custom virtual radios on demand?
- How to model the embedding of heterogeneous RAN slices on RANaaS platforms?
- Could separate specialised orchestrators be used to provide fine-grained resource allocation on E2E networks?
- How to coordinate multiple network segments and orchestrators to guarantee a consistent QoS for E2E NSs?

To addresses these questions, we apply a range of experimental approaches, analytical methods, and new architectural proposals, including the qualitative review of the vast literature on radio virtualisation, radio slicing and RANaaS; the implementation of a software layer that provides radio hypervisors with resource management features for supporting RANaaS; the discussion and formalisation of the radio processing isolation problem; the modelling of the resource allocation of heterogeneous RAN slices on RANaaS platforms; and the proposal of a hierarchical orchestration architecture for E2E networks as well as its implementation as a proof-of-concept prototype.

First, we investigate the functionality and challenges for enabling RANaaS through radio virtualisation. We identify the key requirements for using radio hypervisors to support RANaaS, evaluate how radio hypervisors in the literature meet these requirements, and observe that most of the existing radio hypervisors lack some of the essential features for RANaaS. Then, we introduce a modular software layer that can sit on top of existing radio hypervisors and provide them with the missing slicing functionality to support RANaaS. We integrate our software layer with a radio hypervisor and validate its ability to create and deploy virtual radios with different Radio Access Technologies (RATs) or numerologies on-demand, capable of displaying performance comparable to bare metal.

Next, we focus on the resource management of RANaaS platforms regarding the mapping of real radio resources to realise virtual radios, also known as the virtual wireless network embedding problem. We observe that current models and solutions for the embedding of RAN slices are tied to particular radio virtualisation mechanisms, not being suitable for virtual radios with different RATs or numerologies. Then, we introduce a technology-agnostic modelling approach for embedding heterogeneous RAN slices, which is transparent to the characteristics of the underlying virtual radios. We also propose a resource management optimisation problem that is solved at the MNO, for determining the optimal allocation of RAN slices to minimise the total isolation overhead.

Finally, we look into the coordination of the resource management across multiple network segments to deploy E2E NSs with consistent QoS. We propose a hierarchical orchestration architecture that addresses the limited support and oversimplified resource allocation on different network segments of existing E2E orchestration solutions, by enabling the independent management of each network segment using existing distributed specialised orchestrators used to manage a real network infrastructure. Then, we present a prototype higher-level orchestrator, the hyperstrator, for coordinating the resource management of the existing specialised orchestrators and the deployment of E2E NSs across network segments. We also verify that the distributed nature of our orchestration architecture introduces a negligible overhead for instantiating E2E NSs.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to Prof. Luiz DaSilva. Not only for his continued support and valuable guidance throughout my PhD, but also for believing in me many years ago, giving me a chance to prove myself when I popped by his office asking to become a volunteer at CTVR. Over the years, he gave many incredible opportunities that changed my life, and chances to improve my skills through several trials by fire. I could not have wished for a better teacher, mentor and boss. Last but not least, his most valuable teaching was how to cheat on boardgames like a professional; no game nights in our lab were the same ever again. In addition, I would like to thank Prof. Anil Kokaram for facilitating the submission of my PhD thesis.

To everyone at CONNECT for their unwavering professional and emotional support. I am very grateful to have had the chance of spending the past four years surrounded by many brilliant people, whom not only I have learned a lot from, but that I am now also lucky enough to be able to call friends, including Diarmuid, Conor, Harun, Sandip, and Alan. To Maicon and Merim for the many many hours discussing what is the best text editor, and then shifting as a group between Vim and Emacs countless times. To Davi and Fadhil, my deepest thanks for their trust and friendship, my partners in crime, food, games, and also in our new tech podcast (stay tuned for new episodes). To Nima, whom I had the privilege to serve under as the vice-president of the Coffee Club, albeit he might not be the best food sharer in the world. To Jernej, Susanah, Jacek, and Arman for their friendship in and out of the work environment, regardless if they wanted it or not. And to Natal, the fastest and most reliable coworker ever, the first person to ever make me feel I could not keep up with him. I also would like to thank Andrew for his incredible lessons about communication and presentation, which helped me a lot when disseminating my work, and to Mark and Martin for their continued support and teachings about entrepreneurship and relationship with industry.

A special thanks to everyone in the CoronaQuiz group, the best friends who have always been a great company and source of strength; I simply cannot imagine my life without them (especially

during the last year), including Ferreira, Eric, Cavedon, Marcela, Ana, and Carol. Potentially Ellen as well, but definitely not Daniel; he does not deserve even being mentioned here. Oh, and Laura, the Pacotinha, she is a must. In addition, I would like to thank Maria for being the best pumpkin ever, Anderson for making me get addicted to Minecraft (again), and Chaves, for his continued diminishment of everything I have accomplished so far. Also, I would like to thank Charlie, Aoife, and Conor for their great company, amazing evenings, and most importantly, delicious food. Finally, I need to thank my grandparents, Walter and Tereza, who pretty much made me into what I am today (for better or worse). To Vanessa and Mauro, as well as Erika and Ricardo, who have unconditionally encouraged and supported me throughout the years, and I absolutely could not have wished for better parent figures in my life.

Contents

List of Figures

List of Tables

Dedicated to my late grandparents,
Walter and Tereza Cals, who throughout
their lifetimes etched the importance of
education on the walls of my heart.

Chapter 1

Introduction

Introduction

> “ *Every journey begins with a choice.* ”
>
> Professor Oak, Pokémon Red & Green, *1996*

THIS thesis studies the resource management functionality for enabling End-to-End (E2E) network slicing through radio virtualisation. We propose a solution that leverages radio hypervisors to enable sharing physical network infrastructure, and supports services with diverging performance requirements using virtual radios. We enable the provision of tailored virtual radios with different Radio Access Technologies (RATs) and numerologies in real-time, delivering performance comparable to bare metal. Then, we model the embedding of virtual radios on top of physical radio hardware, and introduce a resource management method that minimises the total overhead required to ensure their independent operation. Furthermore, we present a hierarchical orchestration architecture for managing virtual resources across multiple network segments, and deploying Network Slices (NSs) with consistent Quality of Service (QoS). Figure 1.1 illustrates the scope of this PhD thesis.

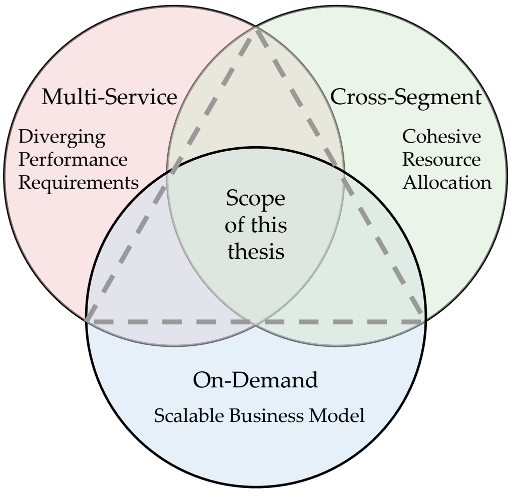


FIGURE 1.1: A diagram of the scope of this PhD thesis.

1.1 Overview

Mobile Network Operators (MNOs) face an accelerating mobile data traffic Compound Annual Growth Rate (CAGR), accompanied by a decrease in their Average Revenue Per User (ARPU), leading to a widely known revenue gap in the mobile network industry [1] [2]. As a result, the MNOs are searching for innovative ways to increase their revenue and ensure profitability, such as providing other communication services beyond Mobile Broadband (MBB) to end-users [3]. In recent years, the MNOs started exploring new markets, e.g., the Internet of Things (IoT) and multimedia broadcast, with the offer of Narrowband IoT (NB-IoT) and Multimedia Broadcast Multicast Service (MBMS), respectively. However, mobile network architectures up to 4G were not designed to support communication services with a wide range of different requirements in terms of performance, scalability and availability [4]. Consequently, there are limitations on the types of communication services that existing mobile network infrastructures can readily accommodate. For example, enabling the support for low-cost, low-power IoT services on top of 4G with NB-IoT was a complex and time-consuming process, which entailed in the development of a new type of RAT, a new release of the LTE standard, and the manufacture of new NB-IoT-compatible base stations [5].

In contrast to previous generations of mobile network architectures, 5G is envisioned from the very beginning to be forward compatible and support a variety of different communication services [5] [6]. As part of its development and standardisation process, the 3rd Generation Partnership Project (3GPP) identified over 50 new potential markets and use cases for 5G that MNOs can target for increasing their revenue [7]. These new communication services fall under four major categories: Enhanced Mobile Broadband (eMBB), Enhanced Vehicular-to-Everything (eV2X), Critical Communications (CriC), and Massive IoT (MIoT), each of which has its own set of challenging and diverging performance requirements [8] [9]. To cope with the requirements of such diverse set of communication services, the 3GPP introduced the concept of network slicing for 5G, which proposes partitioning the physical network infrastructure of an MNO into independent virtual E2E networks, known as NSs [10]. Each NS operates as a completely isolated E2E network that can be individually tailored and configured on the fly for serving different purposes [8]. In this way, network slicing allows MNOs to design and deploy customised NSs on-demand, supporting a wide range of new types of communication services without requiring modifications on the underlying standard or networking equipment [11].

The ability to dynamically deploy customised virtual E2E networks grants versatility to 5G mobile networks [12], enabling new ways that MNOs can monetise their physical network infrastructure and opening unprecedented business opportunities for the mobile network market [13]. For example, MNOs can offer NS as a Service (NSaaS) to business owners with networking demands, providing them with E2E networking solutions in the form of NSs with customised performance and functionality [10], and turning business owners into tenants whose virtual E2E networks seamlessly coexist on top of the MNO's shared physical network infrastructure [14]. Therefore, network slicing breaks the monolithic nature of legacy mobile networks into value-chain networks [15], enabling MNOs to go beyond providing traditional Business-to-consumer (B2C) services to their end-users, and establish a scalable business model by providing Business-to-business (B2B) services to vertical industries, e.g., automotive and healthcare, and Business-to-business-to-consumer (B2B2C) services to service providers, e.g., Mobile Virtual Network Operators (MVNOs) [13] [16].

1.2 Motivation

The E2E network infrastructure of MNOs may comprise multiple network segments, each of which can be independently orchestrated, sliced, and combined for creating different types of NSs [8]. For mobile networks, the 3GPP defines E2E NSs as the combination of Core Network (CN) slices and Radio Access Network (RAN) slices [17], as illustrated in Fig. 1.2. Likewise, the offer of NSaaS can be split into a conjunction of CN as a Service (CNaaS) [18] and RAN as a Service (RANaaS) [19]. Such separation gives MNOs the versatility and granularity to offer tenants: customised CN slices, where each tenant can define its own authentication schemes, mobility and session management, whilst sharing a common RAN among all clients; customised RAN slices, where each tenant can specify its own coverage area, RATs and numerologies, whilst sharing a common CN among all clients; or fully customised E2E NSs, where tenants can define both their CN and RAN [16].

The slicing process is realised by entities known as hypervisors, which are specific to the virtualisation of a particular type of resource and hardware platform [20] [21]. However, while the virtualisation of the CN is a mature area, with carrier-grade virtualisation solutions being adopted in production networks, the virtualisation of the RAN remains a new research topic [21]. The proof-of-concept implementations of radio hypervisors are essential for analysing the benefits of RAN virtualisation, as well as its drawbacks, e.g., performance impacts due to virtualisation overheads, and signal

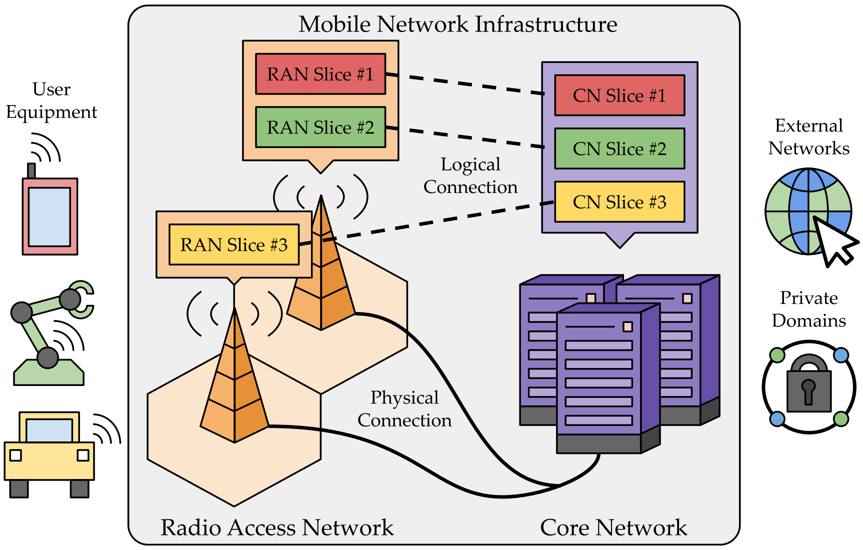


FIGURE 1.2: Example of a mobile network infrastructure comprised of RAN and CN segments, both of which can have different sets of RAN and CN slices logically connected to each other to form E2E NSs. The data from the User Equipments (UEs) traverses through the combination of RAN and CN slices to reach external networks or private domains.

degradations due the isolation between virtual radios [21] [22]. The current prototypes of radio hypervisors available in the literature can create virtual radios for realising RAN slices using either low-level radio resources, e.g., time and frequency; or high-level radio resources, e.g., Physical Resource Blocks (PRBs) and frames [22]. Although paving the way for the realisation of RAN slicing, these prototypes mainly focus on development of scheduling or multiplexing techniques for enabling the coexistence of multiple RAN slices on top of a single physical hardware platform, without considering the additional requirements for RANaaS.

The majority of existing hypervisors operate in a static manner, not allowing the creation, alteration or removal of virtual radios on-demand [23]. In addition, these prototypes lack negotiation and monitoring capabilities for tenants to communicate and query the available radio resources, request RAN slices, and assess the performance of their RAN deployment, which prevents their use for supporting RANaaS [24]. To address these obstacles, in Chapter 3, we identify the requirements for using radio hypervisors to support RANaaS, and we introduce a software layer that can wrap around existing radio hypervisors to provide them with the missing slicing functionality. Another fundamental challenge related to RAN slicing is deciding how to efficiently and optimally map radio resources from the physical radio to realise different RAN slices, also known as the virtual wireless network embedding problem [22]. However, current models and solutions for embedding RAN slices are limited to particular RATs and numerologies, assuming that every virtual radio possesses the same type and granularity of radio resources, which does not hold for heterogeneous RAN slices. Also, these approaches do not factor the additional overhead required to ensure isolation between different virtual radios. In Chapter 4, we introduce a novel modelling approach for embedding heterogeneous RAN slices on RANaaS platforms, which is transparent to the type and granularity of resources of the individual virtual radios, and considers the additional overhead to ensure isolation between RAN slices by design.

We can extend the concept of network slicing to other kinds of E2E network infrastructures, e.g., metro networks from Internet Service Providers (ISPs) or cloud networks from Cloud Providers (CPs), where the E2E NSs consist of a combination of different types of Network Segment Slices (NSSs), e.g., Transport Network (TN) and Data Centre Network (DCN) slices [25]. Regardless of their type, every NSS contributes to the overall QoS of an NS, as the inappropriate resource allocation in a single NSS not only leads to localised bottlenecks but also impairs the performance of all communication services running on the NS [26]. Therefore, guaranteeing consistent QoS for NSs requires cohesive resource allocation across multiple network segments [10] [25]. However, each type of network segment has different paradigms and abstractions, entailing distinct orchestration approaches that require domain expertise [27].

There are specialised orchestrators tailored for the particularities of specific segments, e.g., the Open Network Operating System (ONOS) [28] and the Open-source MANO (OSM) [29], which provide fine-grained resource allocation in the TN and DCN segments, respectively. However, the interaction between different orchestrators for a joint orchestration of multiple network segments in E2E networks remains an open challenge [25]. Conversely, there are one-size-fits-all orchestrators that aim at orchestrating entire E2E networks, e.g., CORD [30] and 5G-EmPOWER [31], which provide a central point of management for the entire E2E infrastructure. However, one-size-fits-all orchestrators tend to oversimplify the particularities of certain network segments and only support limited technologies and standards; being exceedingly complex for adding new functionality, e.g., the state-of-the-art on resource management, or incorporating new types of network segments, e.g., mmWave and satellite networks [32]. To tackle these issues, in Chapter 5, we propose a hierarchical orchestration architecture for E2E networks, leveraging domain expertise and enabling the independent management of each network segment using existing distributed specialised orchestrators already deployed in the network infrastructure.

1.3 Research Questions

This thesis addresses the following Research Questions (RQs):

- What are the requirements for using radio hypervisors to support RANaaS?
- How can radio hypervisors provision and instantiate custom virtual radios on demand?
- How to model the embedding of heterogeneous RAN slices on RANaaS platforms?
- Could separate specialised orchestrators be used to provide fine-grained resource allocation on E2E networks?
- How to coordinate multiple network segments and orchestrators to guarantee a consistent QoS for E2E NSs?

1.4 Contributions

In this section, we summarise the key research contributions of this PhD thesis.

Design and Development of a Technology-agnostic RANaaS Platform – Chapter 3

- Identifying the key requirements for using radio hypervisors to support RANaaS.
- Developing a modular software layer that can sit on top of existing radio hypervisors and provide them with the missing slicing functionality.
- Integrating our software layer with HyDRA [21] for creating and deploying virtual radios with different RATs or numerologies on-demand.
- First work to quantify the delay and signal degradation introduced by radio virtualisation.

Embedding of Heterogeneous Virtual Wireless Networks – Chapter 4

- First work to formulate the embedding of heterogeneous RAN slices using a Travelling Salesman Problem (TSP)-based approach.
- Proposing a resource management optimisation problem for minimising the total isolation overhead between virtual radios.
- Assessing the performance and computation complexity for solving the embedding of heterogeneous RAN slices using different heuristic methods.

Breaking Down the E2E Resource Management and Network Slicing – Chapter 5

- Enabling the independent management of each network segment, leveraging existing distributed specialised orchestrators.
- Developing a prototype higher-level orchestrator, the hyperstrator, for coordinating the resource management of the specialised orchestrators.
- First work to decentralise the control over the physical infrastructure and the decision over the resource management for E2E networks comprised of different network segments.
- Evaluating the overhead introduced by our hierarchical orchestration architecture to provision NSs, and the impact of each NSS on the performance of NSs.

1.5 Thesis Outline

The remainder of this PhD thesis is organised as follows:

Chapter 2 – Background

In Chapter 2, we provide technical background on the concepts used throughout this PhD thesis and review the literature relevant to the scope of our work. First, we examine the concepts of softwarisation, virtualisation and orchestration, using examples of their application in mobile networks. Then, we review the state-of-the-art on radio virtualisation, and categorise existing radio hypervisors with respect to their radio virtualisation mechanisms. Finally, we assess the current cross-network segment orchestration solutions regarding their orchestration characteristics, as well as their support for different technologies and slicing of network segments.

Chapter 3 – Leveraging Radio Hypervisors to Enable RANaaS

In Chapter 3, we investigate the functionality and challenges for enabling RANaaS through radio virtualisation. We analyse the requirements for using radio hypervisors to support RANaaS, and we evaluate how the current state-of-the-art on radio virtualisation meets such requirements. We identify, formalise and address the key resource management functionality missing from existing radio hypervisors. Then, we introduce eXtensible Virtualisation Layer (XVL), a software layer that provides the missing resource management functionality for enabling RANaaS and can be added on top of existing radio hypervisors. We detail the integration of XVL with HyDRA, forming a RANaaS platform that can provision heterogeneous RAN slices as a service. Next, we outline XVL's architecture and design choices, as well as evaluate its performance in terms of the computational overhead, the delay to provision virtual radios, the delay introduced to forward In-phase & Quadrature (IQ) samples, and the signal degradation. Our results show that XVL enables leveraging existing radio hypervisors for supporting the RANaaS paradigm.

Chapter 4 – Embedding Heterogeneous RAN Slices on RANaaS Platforms

In Chapter 4, we focus on the resource management of virtual radios on RANaaS platforms. First, we assess how the current virtual wireless network embedding solutions model the allocation of real radio resources to virtual radios, and demonstrate how they are not suitable for embedding heterogeneous RAN slices. Then, we introduce a novel graph-based modelling approach for embedding heterogeneous RAN slices, and propose a new formulation that considers the additional required guard bands to ensure isolation between RAN slices by design, while also being transparent to the type and granularity of radio resources of the individual virtual radios, and extensible to support new RATs and numerologies. Based on this formulation, we also propose a resource management optimisation problem that is solved at the MNO, for determining the optimal allocation of RAN slices to minimise the total isolation overhead. Finally, we assess different methods for solving the resource management optimisation problem, and compare the computational footprint of the optimal solution against two heuristic algorithms that provide a good solution for the problem while possessing a lower computational footprint.

Chapter 5 – Orchestrating Multiple Network Segments to Create E2E NSs

In Chapter 5, we address the limited support and oversimplified resource allocation on different network segments of existing E2E orchestration solutions. We propose a hierarchical orchestration architecture for E2E networks, breaking down the E2E resource management and network slicing problems per network segment. We introduce a higher-level orchestrator, the hyperstrator, to coordinate existing distributed orchestrators and deploy NSs across multiple network segments in current network deployments. Then, we detail a prototype implementation of the hyperstrator and validate our hierarchical orchestration concept with two proof-of-concept experiments, showing the NS deployment and how the resource allocation per network segment impacts the performance of NSs. The results show that the distributed nature of our orchestration architecture introduces negligible overhead for provisioning NSs in our particular setting, and confirm the need of a hyperstrator for coordinating multiple network segments and ensuring consistent QoS for the NSs.

Chapter 6 – Conclusions and Future Directions

In the last chapter, we conclude this PhD thesis by summarising our key contributions and discussing the outcomes of our research efforts. In addition, we examine future directions for potential extensions to the contributions achieved during the PhD project.

1.6 Dissemination

In this section, we list the dissemination efforts during the PhD project.

Journal Publications

6. Joao F. Santos, Davi da Silva Brilhante, José F. de Rezende, Nicola Marchetti, and Luiz A. DaSilva “Optimal Embedding of Heterogeneous RAN Slices for Technology-agnostic RANaaS”, *IEEE Wireless Communications Letters (WCL)*, under review.

5. Erika Fonseca, Joao F. Santos, Francisco Paisana, and Luiz A. DaSilva, “Radio Access Technology Characterisation Through Object Detection”, *Elsevier Computer Communications (COMCOM)*, accepted for publication, 2020.

4. Maicon Kist, Joao F. Santos, Diarmuid Collins, Juergen Rochol, Luiz A. DaSilva, and Cristiano B. Both, "AIRTIME: End-to-end Virtualization Layer for RAN-as-a-Service in Future Multi-Service Mobile Networks”, *IEEE Transactions on Mobile Computing (TMC)*, accepted for publication, 2020.

3. Joao F. Santos, Wei Liu, Xianjun Jiao, Sofie Pollin, Johann M. Marquez-Barja, Ingrid Moerman, and Luiz A. DaSilva, “Breaking Down Network Slicing: Hierarchical Orchestration of End-to-End Networks”, *IEEE Communications Magazine (ComMag)*, vol. 58, no. 10, pp. 16–22, 2020.

2. Joao F. Santos, Maicon Kist, Juergen Rochol and Luiz A. DaSilva, “Virtual Radios, Real Services: Enabling RANaaS Through Radio Virtualisation”, *IEEE Transactions on Network and Service Management (TNSM)*, vol. 17, no. 4, pp. 2610-2619, 2020.

1. Wei Liu, Joao F. Santos, Jonathan van de Belt, Xianjun Jiao, Ingrid Moerman, Johann Marquez-Barja, Luiz DaSilva, and Sofie Pollin, “Enabling Virtual Radio Functions on Software Defined Radio for Future Wireless Networks”, *Springer Wireless Personal Communications (WPC)*, no. 113, pp. 1579–1595, 2020.

Conference Publications

6. Alvaro Ibanez, Borja Inesta, Manuel Fuentes, David Gomez-Barquero, Diarmuid Collins, and Joao F. Santos, “Single Frequency Networks for 5G Broadcast: A Software Defined Radio Experiment”, *IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, Paris, France, 03-05 Jun. 2020.

5. Hazel Murray, Jerry Horgan, Joao F. Santos, David Malone, and Harun Siljak, “Guide to Implementing a Quantum Coin Scheme”, *IET Irish Signals and Systems Conference (ISSC)*, Letterkenny, Ireland, 11-12 Jun. 2020.

4. Ingrid Moerman, Djamal Zeghlache, Adnan Shahid, Joao F. Santos, Luiz DaSilva, Klaus David, John Farserotu, Ad Ridder, Wei Liu, and Jeroen Hoebeke, “Mandate-driven Networking

Eco-system: A Paradigm Shift in End-to-End Communications”, *6G Wireless Summit (6G SUMMIT)*, Levi, Finland, 17-20 Mar. 2020.

3. Joao F. Santos, Maicon Kist, Jonathan van de Belt, Juergen Rochol and Luiz A. DaSilva, "Towards Enabling RAN as a Service - The Extensible Virtualisation Layer", *IEEE International Conference on Communications (ICC)*, Shanghai, China, 20-24 May 2019.

2. Wei Liu, Joao F. Santos, Xianjun Jiao, Francisco Paisana, Luiz A. DaSilva, and Ingrid Moerman, "Using Deep Learning and Radio Virtualisation for Efficient Spectrum Sharing among Coexisting Networks", *EAI International Conference on Cognitive Radio Oriented Wireless Networks (CROWNCOM)*, Ghent, Belgium, 18-20 Sep. 2018.

1. Danielle Caled, Tiago Salmito, Joao F. Santos, Leandro Ciuffo, José Rezende, and Iara Machado, "The Next Generation of the FIBRE Software Architecture", *SBC Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)*, Belém, Brazil, 15-19 May 2017.

Non-Peer-Reviewed Publications

1. Joao F. Santos, Jonathan van de Belt, Wei Liu, Vincent Kotzsch, Gerhard Fettweis, Ivan Seskar, Sofie Pollin, Ingrid Moerman, Luiz A. DaSilva, "Orchestrating Next-Generation Services Through End-to-End Network Slicing", ORCA White Paper, Oct. 2018.

Standardisation Contributions

1. European Telecommunications Standards Institute, “ETSI TR 103 626: Autonomic Network Engineering for the Self-managing Future Internet (AFI); An Instantiation and Implementation of the Generic Autonomic Network Architecture (GANA) Model onto Heterogeneous Wireless Access Technologies using Cognitive Algorithms", *European Telecommunications Standards Institute*, Tech. Rep., Feb. 2020.

Presentations

6. "Distributed End-to-End Network Slicing across Radio Access, Transport, and Core Networks", presentation at the 32nd Workshop of the Wireless Community – Experimentation Facilities for Wireless Connectivity in Industrial IoT Applications, Webinar, 08 Sep. 2020. Presenter: Joao F. Santos

5. "State-of-the-art in Software-defined Wireless Networking", tutorial at the 32nd Workshop of the Wireless Community – Experimentation Facilities for Wireless Connectivity in Industrial IoT Applications, Webinar, 08 Sep. 2020. Presenter: Prof. Ingrid Moerman.

4. "Towards Enabling RAN as a Service - The Extensible Virtualisation Layer", presentation at the IEEE International Conference on Communications (ICC), Shanghai, China, 21 May 2019. Presenter: Joao F. Santos

3. "Hierarchical Orchestration of End-to-End Networks", presentation at the Wireless Innovation Forum European Summit on Wireless Communications Technologies (WInnComm Europe 2019), Berlin, Germany, 15 May 2019. Presenter: Joao F. Santos.

2. "Virtual Radios, Real Services: End-to-End Resource Orchestration for Network Slicing", keynote at the IEEE WCNC 2nd Workshop on Ultra-High Speed, Low Latency and Massive Connectivity Communication for 5G/B5G, 15 Apr. 2019. Presenter: Prof. Luiz A. DaSilva.

1. "Orchestrating Next-Generation Services Through End-to-End Network Slicing", presentation at the Third Global Experimentation for Future Internet (GEFI) Workshop, Tokyo, Japan, 25 Oct. 2018. Presenter: Dr. Ingrid Moerman.

Chapter 2

Background

Background

> “ *Things are only impossible until they are not.* ”
>
> Jean-Luc Picard, Star Trek: The Next Generation, *1988*

IN this chapter, we provide the background on the broader context of the research challenges addressed by this PhD thesis. We examine the fundamental concepts of softwarisation, virtualisation and orchestration, and review the literature in the research areas relevant to this PhD thesis, regarding radio virtualisation and E2E network orchestration.

In Section 2.1, we lay out the theoretical dimension of softwarisation, virtualisation, and orchestration, clarifying the differences and relationship between these concepts, and outline the roles of hypervisors, orchestrators, and controllers used throughout this thesis. In Section 2.2, we review the literature about radio virtualisation as the technological enabler of RAN slicing, and categorise existing radio hypervisors with respect to their radio virtualisation mechanisms. Finally, in Section 2.3, we assess the state-of-the-art on the orchestration of E2E networks, and evaluate existing solutions regarding their orchestration characteristics, support for different technologies, and slicing of network segments.

2.1 Softwarisation, Virtualisation and Orchestration

In this section, we discuss the fundamental concepts of softwarisation, virtualisation, and orchestration, clarifying their differences and relationships. In addition, we outline the roles of hypervisors, orchestrators, and controllers in the context of mobile networks.

2.1.1 Softwarisation

According to the principle of computational equivalence between hardware and software [33]: hardware and software platforms are logically equivalent, and either of them can realise any computing operation. For example, the training of Artificial Neural Networks (ANNs), the routing of network packets, or the (de)mapping of symbols to bits are operations that can be built directly into the hardware, e.g., Application-Specific Integrated Circuit (ASIC), or run as software on General-purpose Processors (GPPs), e.g., Advanced RISC Machines (ARM) [34]. However, such operations do not perform the same in both computing platforms, as there are intrinsic trade-offs between the performance of hardware, regarding execution speed and energy efficiency, and the flexibility of software, regarding versatility and reconfigurability [35]. This dichotomy motivated the development of hybrid hardware-software platforms for achieving different levels of compromise between performance and flexibility [36] [37], bringing certain reconfigurability to hardware with programmable-logic chips, e.g., Field-Programmable Gate Arrays (FPGAs), or accelerating specific software instructions with programmable processors, e.g., Digital Signal Processors (DSPs) [38], [39], as depicted in Fig. 2.1.

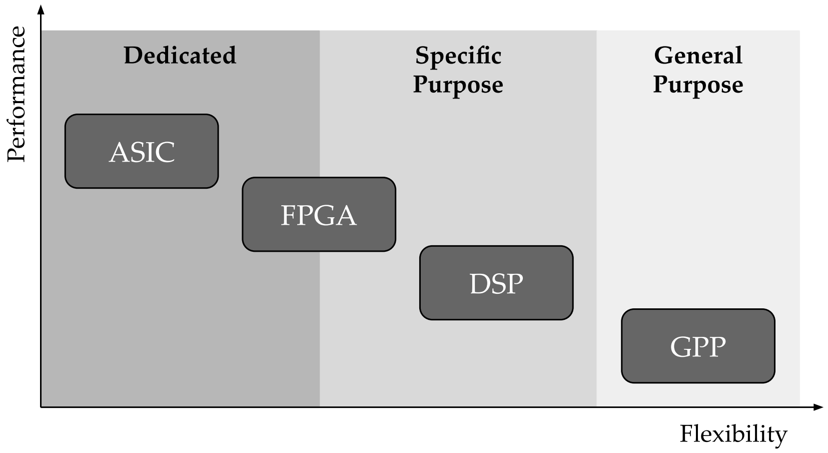


FIGURE 2.1: An illustration of the trade-offs between different computing platforms, regarding performance and flexibility [40].

Mobile networks are becoming increasingly dynamic and context-aware, reacting and adapting to sudden changes in network traffic, demand and health [41]. This movement towards self-organisation calls for reconfigurability over the network functionality, leading to the exploitation of software-based technologies gaining popularity in recent years [42]. The term *softwarisation* refers to the process of realising computing operations that previously were mainly carried out in dedicated hardware, now through software running on top of GPPs [22] [42] [43]. Some notable examples of softwarisation in mobile networks are software-defined hardware platforms such as Software-defined Switches (SDSs) and Software-defined Radio (SDR), which allow the decomposition and reprogrammability over the switching and radio functionality, respectively [44]. SDSs can realise different packet processing and routing protocols in software, not limited to vendor-specific implementations on dedicated hardware [45]. The use of SDSs facilitated the development of tailored networking solutions to cater to business needs [46], and enabled Software-defined Networking (SDN) technology for network control and management [47]. SDRs can realise different digital signal processing chains in software, not limited to a particular RAT as conventional radio platforms [12]. The use of SDRs shortened the testing and prototyping cycles for wireless networks [48] and enabled the development of cognitive radio systems [49].

2.1.2 Virtualisation

The network equipment used in mobile networks possess a set of real resources, i.e., limited attributes or capabilities, required to achieve a given purpose [9]. For example, servers have memory and storage used for computing operations, switches have Ethernet ports and routing tables used for routing network traffic, and radios have Radio Frequency (RF) front-ends and bandwidth used to transmit and receive wireless signals. Usually, such resources are statically tied to the underlying physical network infrastructure and cannot scale to serve variations in demand over time. As a consequence, MNOs often need to overdimension their network deployments to attend peak demand by overprovising network resources, which increases costs and decreases efficiency [50]–[52]. The term *virtualisation* refers to the process of creating a virtual representation of the real resources in software, which can be used in the same way as their real counterparts, but are more easily managed and dynamically tailored to serve different purposes and levels of demand [22] [53] [54]. These virtual resources are indistinguishable from real resources, giving their users the illusion of being in possession of real resources, which allows the creation of different levels of representation and the recursion of virtualisation, e.g., running containers inside a Virtual Machine (VM) [55].

The virtualisation process is realised by entities known as hypervisors, responsible for the abstraction between real and virtual resources, decoupling their use from the underlying physical network infrastructure; and the isolation between virtual resources, preventing them from interfering with one another in harmful ways [51] [53]–[56]. The abstraction may involve different types of resource

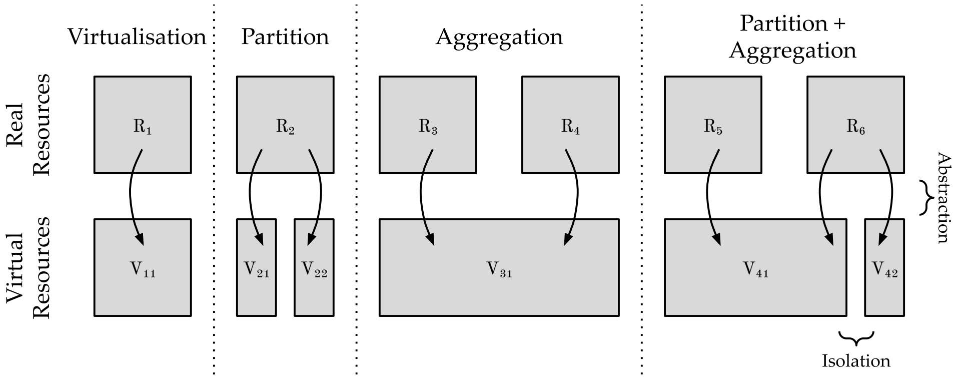


FIGURE 2.2: An example of the abstraction between and virtual resources, showing the different types of resource mapping that hypervisors can perform [22].

mapping: (*i*) the partition of a real resource to form virtual resources; (*ii*) the aggregation of real resources into a virtual resource; and (*iii*) the combination of both partition and aggregation [51] [53], as illustrated in Fig. 2.2. For example, the Linux kernel's Logical Volume Manager (LVM) acts as a storage hypervisor [57], partitioning real storage volumes, e.g., hard disk and solid-state drives, into virtual storage volumes that can be dynamically moved and resized at runtime; and/or aggregating different real storage volumes into a virtual storage volume with larger space and a single standardised interface. The LVM also ensures that any operation realised on the virtual storage volumes does not impair the performance of each other or compromise the underlying physical storage hardware. In that regard, network slicing can be considered a subset of network virtualisation, encompassing only the partitioning of real network resources for creating customised virtual E2E networks that seamlessly coexist on top of a shared physical network infrastructure [9] [22].

2.1.3 Orchestration

Mobile networks incorporate many different technologies from specialised segments, e.g., radio, transport, and core networks [58]. Such complexity makes the design, deployment and operation of new communication services a costly and time-consuming process in production networks, which can take weeks or months of planning, configuring and testing through traditional Operation Support Systems (OSSs) [58] [59]. However, softwarisation and virtualisation introduce new ways for efficient and flexible utilisation of the mobile network infrastructure, allowing MNOs to simplify and automate the resource management, ultimately reducing the time and costs associated with the instantiation of communication services [59] [60]. The term *orchestration* refers to the process of automatically coordinating, configuring, and monitoring the network resources, equipment, and applications to meet a particular objective [58]–[60]. It involves the translation of high-level service requests with requirements, e.g., mobile coverage in a given geographical location, into the necessary low-level network configuration to fulfil these requirements, e.g., set of Remote Radio Heads (RRHs) with a particular amount of PRBs; as well as the interaction with the underlying physical network infrastructure for applying the configuration on the network equipment [58]–[60].

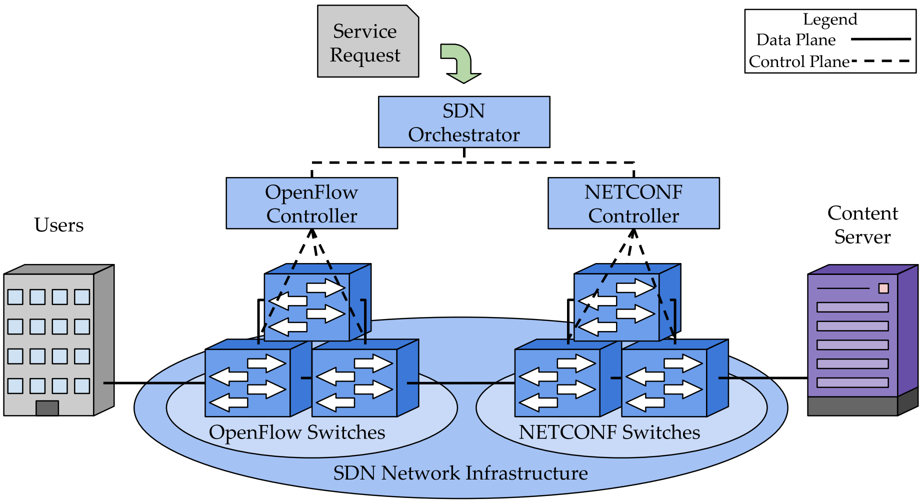


FIGURE 2.3: An example of an orchestrator coordinating the resources of a physical SDN infrastructure to fulfil service requests, leveraging different specialised controllers to control OpenFlow and NETCONF SDSs [61].

Orchestrators are responsible of processing service requests, implementing the control logic behind the requirement translation, and coordinating the operation of network equipment [58]–[60]. For example, ONOS [28] is an SDN orchestrator that (*i*) responds to intents, high-level service requests detailing an objective, e.g., balance the load, create a firewall, or establish communication between servers; (*ii*) translates them into the necessary network configuration, e.g., packet matching and routing rules; and (*iii*) interfaces with the underlying SDN network infrastructure, e.g., through OpenFlow or NETCONF protocols [58] [59]. A controller often intermediates the communication between the orchestrator and the physical network infrastructure, centralising the control over a set of similar network equipment, which simplifies the realisation of the orchestrator [25] [58]. Each type of controller is designed to control and abstract a particular type of physical hardware platform [62], e.g., the OpenStack controllers manage compute nodes in DCNs, and the Radio Network Controllers (RNCs) manage NodeBs in 3G networks. In this way, network deployments consisting of different network equipment, or using different communication protocols, usually employ a combination of specialised controllers [61], as illustrated in Fig. 2.3. Some controllers may act as hypervisors and, virtualises certain programmable hardware platforms, e.g., the OpenFlow controller FlowVisor [20].

Softwarisation, virtualisation and orchestration enable unprecedented reconfiguration and automation over the mobile network infrastructure, playing a key role in the realisation of network slicing [43] [63] [64]. The utilisation of orchestrators, controllers, and hypervisors greatly simplifies network management and operation, allowing MNOs to automate the deployment of new communication services and offer it as a service [63] [64].

2.2 Radio Virtualisation

Radio hypervisors enable shared RAN deployments, where multiple tenants share a common physical network infrastructure, decreasing the deployment costs and the amount of underutilised radio resources. Some radio hypervisors can support heterogeneous RAN deployments, enabling a single physical radio to realise multiple RATs or numerologies, reducing the number of physical radios needed for deploying heterogeneous mobile networks. Some radio hypervisors can support heterogeneous RAN deployments, enabling a single physical radio to realise multiple RATs or numerologies, reducing the number of physical radios needed for deploying heterogeneous mobile networks. This sharing of infrastructure enabled by radio virtualisation can lead to cost reductions of significant value for MNOs serving special-purpose RATs [24], or operating in Citizens Broadband Radio Service (CBRS) and Unlicensed National Information Infrastructure (U-NII) bands that can accommodate a number of different 3GPP and non-3GPP RATs, e.g., LTE, 5G and WiFi [65]. Fig. 2.4 illustrates two scenarios in which a radio hypervisor manages the radio resources for creating different RAN slices. Virtual Radio [24], one of the initial works on radio virtualisation, argued for the deployment of software-based base stations, with a tailored amount of radio resources and protocol stack to mitigate the slow development cycle in the wireless network industry. Ultimately, the work of [24] proposes the use of radio virtualisation for enabling the deployment of virtual wireless networks on demand, years before the term RANaaS was coined [19]. The following research efforts on radio virtualisation focused on enabling radio slicing by ensuring isolation at the radio resource level [22]. The next natural step is to leverage the capabilities of current radio hypervisors for provisioning RAN slices on-demand, towards enabling the RANaaS paradigm.

There is a vast literature on radio virtualisation, radio slicing and RANaaS [9] [43] [66]. However, the majority of current research efforts in these areas have been theoretical, while the number of experimental and prototype radio virtualisation platforms pales in comparison [22]. Among the

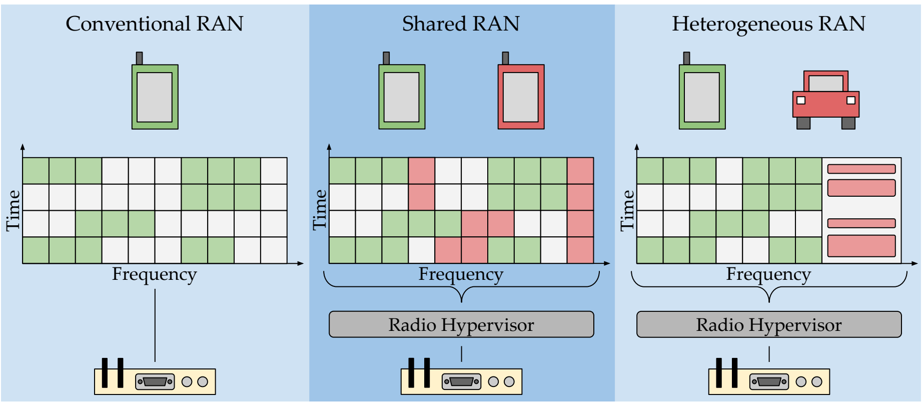
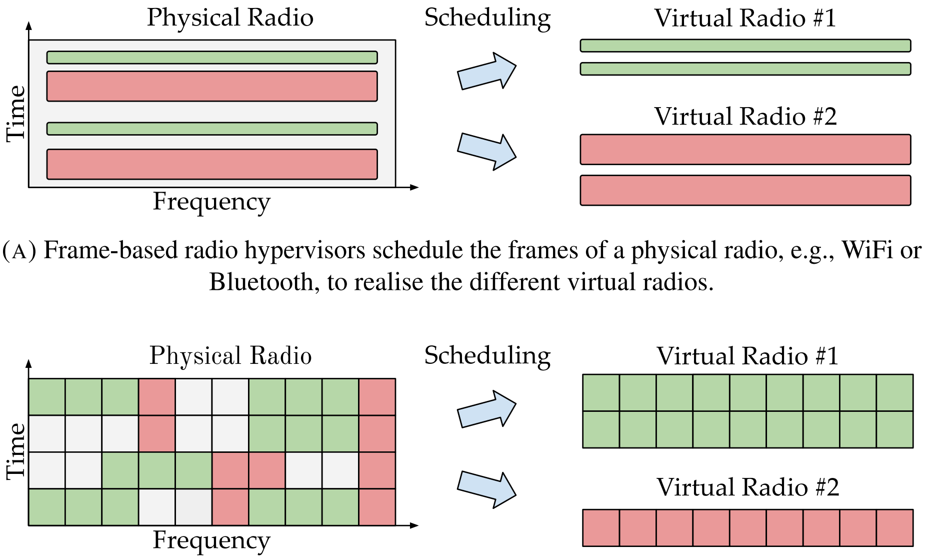


FIGURE 2.4: Radio hypervisors enable sharing of the physical network infrastructure by multiple tenants and/or the use of a single hardware platform to realise heterogeneous RATs. The different colours (green and red) indicate RAN slices belonging to different tenants.

existing experimental research efforts, their proof-of-concept radio hypervisors may differ in terms of approaches to radio virtualisation. We can classify the existing radio hypervisors into two main categories: technology-specific and general-purpose radio hypervisors. In this section, we describe each of these classes, outline their differences, and list some notable examples in the literature.

2.2.1 Technology-specific Radio Hypervisors

Most of the existing radio hypervisors focus on the virtualisation of a single RAT, leveraging the resource allocation capabilities of the given RAT for creating virtual radios. This class of radio hypervisors typically operates at the MAC layer, virtualising physical radios by scheduling the use of their radio resources to different virtual radios. This scheduling-based virtualisation approach supports a single common PHY, shared among all virtual radios, only allowing the creation of virtual radios that use the same RAT as the underlying physical radio [22]. There are two subclasses of technology-specific radio hypervisors, based on the type of MAC of the targeted RAT [66]: frame-based and grid-based radio hypervisors, illustrated in Fig. 2.5. The former targets the virtualisation of RATs that transmit intermittent frames with random contention windows, allocating resources as a number of consecutive frames, e.g., WiFi and Bluetooth. These radio hypervisors operate scheduling airtime, frames and/or different contention window durations to different virtual radios, queueing the frames from all the virtual radios [67]–[69]. Grid-based radio hypervisors target the virtualisation of RATs that transmit continuously, allocating resources on a well-defined grid of resources blocks, e.g., LTE and WiMax. These radio hypervisors operate scheduling Virtual Resource Blocks (VRBs) to the virtual radios, which are non-overlapping subsets of the underlying PRBs [70]–[72].



(A) Frame-based radio hypervisors schedule the frames of a physical radio, e.g., WiFi or Bluetooth, to realise the different virtual radios.

(B) Grid-based radio hypervisors schedule the PRBs of a physical radio, e.g., LTE or WiMax, to realise the different virtual radios.

FIGURE 2.5: Different types of technology-specific radio hypervisors and their virtualisation mechanisms. This class of radio hypervisors operates at the MAC layer, scheduling the available real radio resources to the virtual radios.

2.2.2 General-purpose Radio Hypervisors

There are few examples of radio hypervisors that employ technology-agnostic radio virtualisation mechanisms and support virtual radios with different RATs. This class of radio hypervisors operates at the PHY layer, virtualising the physical radios by multiplexing the use of their RF front-ends. This multiplexing-based virtualisation approach enables each virtual radio to have its own RAT, with independent PHY and MAC. However, general-purpose radio hypervisors require full control over the processing of IQ samples on the physical radio, and hence, can only operate on top of SDR platforms. There are two subclasses of general-purpose radio hypervisors, based on their multiplexing mechanism: FFT-based [21] [73] and filterbank-based [74] radio hypervisors. In both cases, the radio hypervisor partitions the real RF front-end in the frequency domain, creating virtual RF front-ends using non-overlapping spectrum bands, illustrated in Fig. 2.6. Then, the radio hypervisor provides each virtual radio its own virtual RF front-end, which the virtual radios can use to realise their own RAT. However, the ability of being technology agnostic comes at the price of reduced resource efficiency in comparison to technology-specific radio hypervisors, as guard bands and/or intervals are often necessary to ensure isolation between the virtual radios and avoid interference from the Out-of-band Emission (OOBE) of different RATs or numerologies [22].

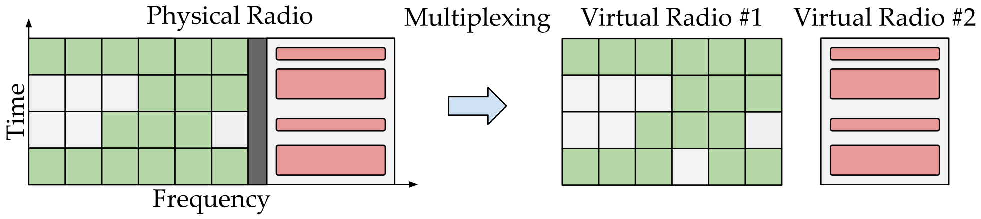


FIGURE 2.6: General-purpose radio hypervisors operate at the PHY layer, multiplexing the RF front-end of an SDR to realise virtual radios with different RATs or numerologies, including the required additional guard band (dark grey).

Existing proof-of-concept radio hypervisors tend to focus on developing radio virtualisation mechanisms, improving the efficiency of the scheduling and multiplexing operations, as well as the isolation between virtual radios. However, these radio hypervisors do not explore broker mechanisms for interacting with third parties, implement the features required for creating virtual radios at runtime, nor examine additional requirements for supporting services such as RANaaS [23] [14]. These issues motivated us to develop a software layer that wraps around existing radio hypervisors and provides them with the missing resource management functionality for RANaaS, discussed in Chapter 3. It is also worth mentioning that one of the main limitations of technology-agnostic radio virtualisation has been the limited availability and potentially prohibitive costs of SDRs with ultra-wide bandwidth (equal or greater than 500 MHz), for allowing the radio hypervisor to support RATs that are very far apart in frequency, or accommodating multiple instances of 5G virtual radios (each of which can occupy up to 100 MHz), all on the top of a shared RF front-end. Such bandwidth limitation can potentially constrain the applicability of this type of radio hypervisor to RATs operating in spectrum bands closer to each other, and reduce the number of virtual radios support simultaneously. However, we do not believe that the current limitations of existing radio hardware diminish the benefits in radio resource management brought by the unique flexibility of technology-agnostic radio hypervisors.

2.3 End-to-End Network Orchestration

In the literature, there are a number of open-source initiatives that provide one-size-fits-all solutions for managing E2E networks. However, their orchestration approaches may differ regarding: (*i*) the degree of centralisation of their control and decision functionality, i.e., centralised or distributed;

(*ii*) the level of disaggregation of their orchestration logic, i.e., implemented by the orchestrator itself or by external applications; and (*iii*) their support for different administrative domains, network segments and technologies. In the remainder of this section, we review some notable examples of one-size-fits-all orchestrators and compare their different approaches to network orchestration. Finally, by analysing these solutions, we identify the missing gaps in the state-of-the-art on E2E network orchestration.

The CORD orchestrator [30] aims at transforming the functionality of central offices, from traditional facilities that provide legacy services using dedicated hardware, into agile DCNs following the Everything as a Service (XaaS) paradigm [30]. CORD is particularly popular among ISPs and MNOs, supporting the instantiation of enterprise Content Delivery Networks (CDNs) and Virtual Private Networks (VPNs), as well as Baseband Units (BBUs) and Evolved Packet Cores (EPCs). CORD realises communication services as chains of functions, using Network Function Virtualisation (NFV) to achieve fast and flexible deployment, and SDN to interconnect Physical Network Functions (PNFs), e.g., physical servers and switches; and Virtual Network Functions (VNFs), e.g., load balancers and firewalls [30]. However, CORD possesses limited capabilities regarding wireless network segments, only supporting Centralised-RAN (C-RAN) LTE deployments.

Similar projects, e.g., 5G-EmPOWER [31], Kista orchestrator [75] and ONAP [76], also leverage NFV and SDN to realise communication services as chains of functions on shared Commercial-Off-The-Shelf (COTS) computing hardware. The former, 5G-EmPOWER, focuses on orchestrating E2E networks with multiple RATs. It is compatible with a wider variety of wireless network segments, supporting Distributed-RAN (D-RAN) deployments of LTE and Wi-Fi RATs, realised on SDRs and embedded Linux devices, respectively [31]. In contrast, the Kista orchestrator focuses on orchestrating E2E networks using C-RAN and different types of TN segments. It supports optical-based TNs for the fronthaul between BBUs and RRHs, and packet-based TNs for the backhaul between BBUs and data centres [75]. While the latter, ONAP, is a comprehensive orchestration platform backed by The Linux Foundation [77], focusing on the orchestration of carrier-grade data centres and mobile networks. It supports the deployment of E2E NSs across different administrative domains, using a combination of the local cloud and edge resources. These three orchestrators disaggregate the orchestration logic from their platforms, not implementing any resource management directives, i.e., the intelligence behind the resource allocation and function placement. Instead, these solutions expect network applications, i.e., custom-made or third-party plugins, to manage the entire E2E network infrastructure. This approach facilitates MNOs to programmatically define the behaviour of their networks, at the cost of requiring further development before initial usage.

The previous one-size-fits-all orchestrators interface with the underlying physical network infrastructure of a single administrative domain through distributed controllers. Each controller is responsible for carrying out resource allocation and function placement tasks on a specific type of network segment, simplifying the interaction between the orchestrator with heterogeneous hardware platforms [25]. However, these orchestrators take the opposite stance regarding their decision functionality, i.e., the intelligence behind the resource negotiation and management for the entire E2E network, centralising it in a single monolithic entity. This approach leads to complex and tightly integrated implementations, which makes including new functionality and supporting new types of network segments cumbersome and non-trivial [32]. Based on these issues, the conceptual work of [32] proposed a hierarchical orchestration architecture for coordinating NFV-based DCNs across multiple administrative domains. This paradigm was later standardised by ETSI [78], which proposed a higher-level NFV orchestrator, the NFVO Composite (NFVO-C), for decomposing service requirements and coordinating distributed lower-level entities, multiple instances of the NFVO Nested (NFVO-N), responsible for the resource management in their own data centres. However, these solutions only focus on orchestrating DCNs, not supporting managing the resources or slicing any type of wireless network segments. Table 2.1 summarises the results of our investigation, considering different orchestrator approaches, supported technologies, and slicing of network segments.

Mobile networks are evolving and incorporating new types of wireless network segments for serving current and future use cases, e.g., the addition of mmWave links, and the inclusion of satellite links for ubiquitous connectivity beyond 5G. MNOs will need to integrate these new segments with their existing network deployments, efficiently orchestrating heterogeneous resources across their extended E2E infrastructure. However, we observe that existing one-size-fits-all orchestrators are not suitable for such scenarios, due to their ossified centralised E2E network management, and limited support for wireless network segments, with a coarse-grained placement of radio functionality that may lead to suboptimal E2E performance [79]. These issues motivated us to develop a hierarchical orchestration scheme detailed in Chapter 5, for enabling the independent management of each network segment by specialised orchestrators, coordinated through a higher-level entity, the hyperstrator.

	Orchestration Approaches			Supported Technologies				Network Segment Slicing			
	Control Centralisation	Decision Centralisation	Logic Disaggregation	LTE	WiFi	SDN	NFV	RAN	TN	CN	DCN
CORD [30]	Distributed	Centralised	Internal	C-RAN		✓	✓	✓	✓	✓	
5G-EmPOWER [31]	Distributed	Centralised	External	D-RAN	D-RAN	✓	✓	✓	✓		
Kista Orchestrator [75]	Distributed	Centralised	External	D-RAN		✓	✓	✓	✓		✓
ONAP [78]	Distributed	Centralised	External	C/D-RAN		✓	✓	✓	✓	✓	✓
ETSI NFVO-C [78]	Hierarchical	Distributed	Internal				✓			✓	✓
Hyperstrator [Chapter 5]	Hierarchical	Hierarchical	Internal	C/D-RAN	D-RAN	✓	✓	✓	✓	✓	✓

TABLE 2.1: Qualitative comparison of characteristics and features present in existing E2E network orchestrators.

2.4 Conclusions

In this chapter, we have presented the concepts of softwarisation, virtualisation and orchestration, as well as detailed the purpose of hypervisors, orchestrators, and controllers in the context of mobile networks. Then, we provided a literature review on the current state of radio virtualisation and E2E network orchestration. Although current radio hypervisors can use a single physical radio to deploy heterogeneous RAN slices with different RATs and numerologies, these radio hypervisors lack certain resource management functionality for deploying virtual radios on-demand, preventing their use to support RANaaS. Moreover, we observe that existing one-size-fits-all orchestrators can coordinate different network segments and types of resources for deploying communication services. However, these orchestrators possess limited capabilities regarding their support for wireless network segments, essential for deploying E2E NSs in mobile networks. This issue is aggravated by the monolithic and tightly integrated implementation of such orchestrators, restricting the inclusion of new functionality and support for new network segments.

Chapter 3

Leveraging Radio Hypervisors to Enable RANaaS

Leveraging Radio Hypervisors to Enable RANaaS

> “*Just because something works,*
> *doesn't mean that it cannot be improved.*”
>
> Shuri, Black Panther, *2018*

IN this chapter, we address the challenges associated with the resource management functionality for enabling RANaaS through a radio virtualisation solution. We start by providing a qualitative evaluation of the required features radio hypervisors to support RANaaS. Then, we introduce the eXtensible Virtualisation Layer (XVL), a software layer that wraps around radio hypervisors and provides them with the missing resource management functionality to be suitable for RANaaS. Next, we formalise and address the isolation of virtual radios at the radio processing level. The combination of our software layer with a radio hypervisor results in a prototype RANaaS platform, as illustrated in Figure 3.1. To the best of our knowledge, our RANaaS platform is the first to enable the provisioning of heterogeneous virtual radios as a service, isolated both at radio resource and radio processing levels, including a monitoring interface for tenants to query the performance of their RAN slices. Finally, we confirm the low computational footprint for running our software layer alongside a radio hypervisor, enabling their utilisation on top of COTS computing hardware.

The technical work presented in this chapter is based on our papers “Towards Enabling RAN as a Service - The Extensible Virtualisation Layer” presented at IEEE International Conference on Communications (ICC) 2019 [80], and “Virtual Radios, Real Services: Enabling RANaaS Through Radio Virtualisation” published in IEEE Transactions on Network and Service Management [81].

The remainder of this chapter is organised as follows. In Section 3.1, we identify the resource management requirements for using radio hypervisors to support RANaaS, evaluate how existing radio hypervisors meet these requirements, and position our work with respect to them. In Section 3.2, we introduce a software layer that implements the missing resource management functionality and can be added on top of existing radio hypervisors, allowing them to provision virtual radios as a service. In Section 3.3, we formalise and address the challenge of isolating virtual radios at the radio processing level. In Section 3.4, we describe the integration of our software layer with a radio hypervisor, evaluate its performance in different scenarios, and validate its capability to serve as an enabler for RANaaS. Finally, in Section 3.5, we conclude this chapter and pose our final remarks.

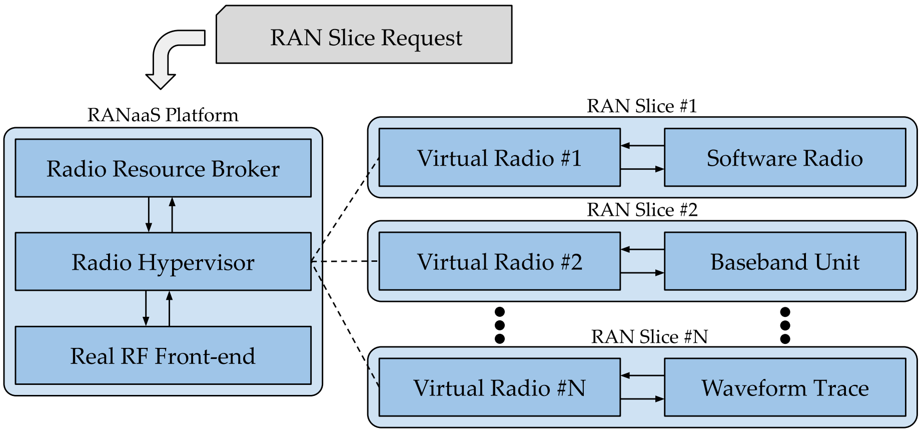


FIGURE 3.1: Functional example of a RANaaS platform leveraging a radio hypervisor and supporting the creation of RAN slices on-demand. The radio hypervisor slices the real RF front-end and creates isolated virtual radios. Each RAN slice has a virtual radio and a source/destination for streams of IQ samples, e.g., produced by software radios, BBUs, or waveform traces.

3.1 Enabling RAN as a Service Through Radio Virtualisation

Inspired by the initial concepts in Virtual Radio [24] and further works on the virtualisation of wireless networks [22] [54] [66] [82], we now examine the use of radio virtualisation for enabling the RANaaS paradigm. In this section, we devise a comprehensive list of required resource management features for a RANaaS platform leveraging radio virtualisation. Then, we assess how well the radio hypervisors listed in Section 2.2 meet these requirements, and position our solution, the XVL, in relation to them.

3.1.1 Requirements for Supporting RANaaS

The MNOs providing RANaaS may charge tenants for using their radio resources according to the spectrum band, service duration, or Service Level Agreement (SLA), based on auctions or other business models [83]. In addition, the type and availability of radio resources may vary across different geographical locations. Therefore, the tenants must be able to discover the available radio resources and negotiate their use [24]. This interaction between radio hypervisors and tenants requires a radio resource broker with an interface for responding to remote requests. Such an interface must have a well-defined syntax for the communication with tenants, and a clear model for the description of the available radio resources, location and cost [14]. Upon successfully leasing the radio resources, the radio hypervisor must be able to instantiate virtual radios on-demand, as the tenants may start or halt the operation of their RAN slices at will.

The offer of RANaaS should support different virtual wireless networks, not limiting itself to a single technology or protocol [74]. Therefore, the underlying radio hypervisor must employ a technology-agnostic radio virtualisation mechanism to support the creation of virtual radios with different RATs or numerologies. In addition, the virtual radios on a RANaaS platform send/receive IQ samples to/from tenants over network connections, which are subject to variations in latency, jitter and reliability. These variations may affect the timing for producing/consuming IQ samples to/from the radio hypervisor, delaying or interrupting the series of multiplexing operations (detailed further in Section 3.2). Therefore, aside from ensuring isolation in terms of radio resources, preventing interference between virtual radios, the radio hypervisor must also ensure isolation in terms of their radio processing, preventing delays in the processing any IQ streams to affect other virtual radios.

Realising a communication service through a RAN slice on a RANaaS platform introduces a new layer of complexity and another point of failure in the service stack. The radio resources previously dedicated to realising a single service and its RAT, now constitute a pool of resources shared amongst a number of virtual radios. Hence, the tenants must be able to assess the performance of their virtual radios for ensuring the proper operation of their RAN deployment and established SLAs [82]. For instance, tenants should be able to query Key Performance Indicators (KPIs) of their virtual radios, e.g., buffer sizes, processing delay, and SINR degradation. Furthermore, the prototype implementation of the RANaaS platforms should ideally be made available to tenants, for transparency and security [54]; and to the research community, for further research and development.

Based on the aforementioned requirements, below we summarise the necessary features for a RANaaS platform leveraging radio virtualisation:

- Resource Negotiation: to possess an interface for querying and requesting RAN slices, describing both the radio resources, e.g., in terms of centre-frequency and bandwidth; and the virtual radios, e.g., owner, transmitter and/or receiver capabilities.
- Dynamic Allocation: to allow the creation and destruction of virtual radios on demand, without interrupting the operation of the radio hypervisor or other virtual radios.
- Technology Agnostic: to employ radio virtualisation mechanisms that support different RATs, and do not limit the choice of the PHY and MAC layers of the RAN slices.
- Radio Resource Isolation: to allocate non-overlapping radio resources to different virtual radios, and prevent interference between virtual radios, e.g., through filtering, guard bands and/or intervals.
- Radio Processing Isolation: to ensure that virtual radio instances cannot affect the operation and performance of each other, even in case of a malfunctioning or misbehaving virtual radio.
- Service Monitoring: to collect and provide information about the performance of individual virtual radios, e.g., in terms of buffer sizes, the introduced delay, and SINR degradation.

The radio hypervisors that meet these resource management requirements can act as RANaaS platforms, enabling MNOs to decrease their deployment costs and introduce new sources of revenue [80]. However, despite the capabilities of existing radio hypervisors for enabling multiple heterogeneous virtual radios to share the same underlying physical radio hardware, not all types of radio hypervisors are suitable or possess the complete set of required features for supporting RANaaS, as discussed in the next section.

3.1.2 State-of-the-Art on Radio Virtualisation for RANaaS

We have evaluated the notable examples of radio hypervisors mentioned in Section 3.1.1 regarding the requirements for supporting RANaaS, and Table 3.1 summarises the results of our analysis. In general, the majority of research efforts on RAN slicing and radio virtualisation either only focus on particular RATs, or do not make their implementation available as open-source [66].

Radio Hypervisor	Resource Negotiation	Dynamic Allocation	Technology Agnostic	Radio Resource Isolation	Radio Processing Isolation	Service Monitoring
Virtual WiFi [67]	–	✓	–	✓	–	–
CrowdLab [68]	–	–	–	✓	–	–
AMPHIBIA [69]	–	–	–	✓	–	–
FLEXRAN [70]	–	–	–	✓	–	–
NVS [71]	–	–	–	✓	–	–
Orion [72]	✓	✓	–	✓	✓	–
SVL [73]	–	–	✓	✓	–	–
IQ Switch [74]	–	–	✓	✓	–	–
HyDRA [21]	–	–	✓	✓	–	–
XVL [this chapter] + Radio Hypervisor	✓	✓	✓	✓	✓	✓

TABLE 3.1: Qualitative evaluation of the resource management features necessary for radio hypervisors to support RANaaS.

One of the few works that is technology agnostic and provides an actual open-source implementation is HyDRA [21], a radio hypervisor for SDRs developed on top of GNU Radio [84], a Software Development Kit (SDK) that provides signal processing blocks for the realisation of RATs and signal-processing systems (detailed further in Section 3.4.1). HyDRA is an FFT-based general-purpose radio hypervisor, where each virtual radio can realise any RAT using its own virtual RF front-end. This radio hypervisor allows multiple RAN slices with heterogeneous RATs to share the same underlying physical radio hardware. However, its virtual radios are not isolated in terms of radio processing, given that if any of the virtual radio delays transmitting/receiving or crashes, it will halt the operation of the radio hypervisor and all other virtual radios. Moreover, the number of virtual radios and their resource allocation, i.e., the amount of spectrum for each virtual RF front-end, must be set up before the operation of the radio hypervisor, as it is not possible to (de)allocate virtual radios without interrupting their operation. As shown in Table 3.1, both HyDRA and the majority of other existing radio hypervisors are still missing essential features for supporting the RANaaS paradigm. Nonetheless, given its capabilities, reliability, and open-source nature, in this chapter we adopt HyDRA as the starting point in the development of a prototype RANaaS platform.

3.2 Extensible Virtualisation Layer

We developed XVL, a cross-platform software layer that sits on top of existing radio hypervisors. It works as a wrapper, leveraging the radio slicing capabilities of an underlying radio hypervisor, and providing it with the missing resource management functionality for supporting RANaaS. This modular design allows the radio hypervisor to focus on the radio virtualisation, offloading to XVL the majority of other tasks, e.g., communication interface, resource allocation, and service monitoring. In the following subsections, we detail how XVL addresses the limitations and missing features identified in Section 3.1.1. Namely, we discuss how XVL: (*i*) provides a communication interface for resource negotiation and allocation; (*ii*) supports the creation and destruction of virtual radios on-demand, (*iii*) ensures radio processing isolation; and (*iv*) monitors the performance of virtual radios.

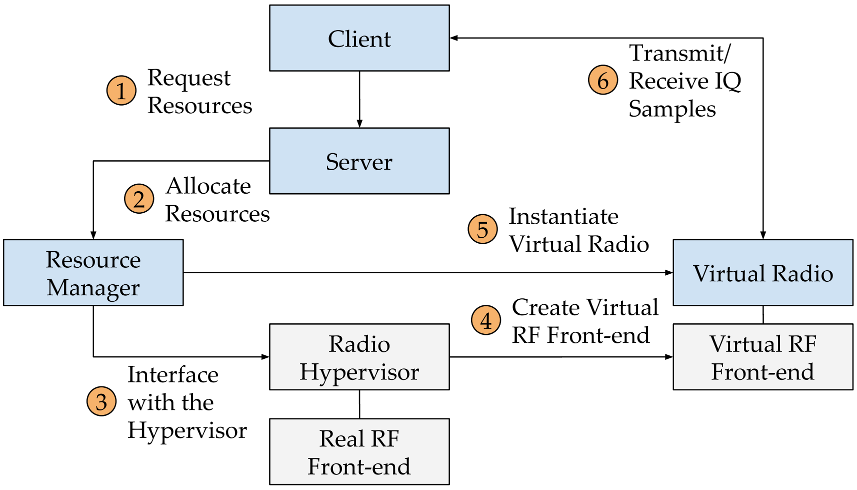


FIGURE 3.2: The steps involved for negotiating the use of radio resources and allocating new virtual radios. XVL (blue) stands as a wrapper around the radio hypervisor (grey).

3.2.1 Resource Negotiation and Dynamic Allocation

XVL operates on a client-server paradigm, with a single server, i.e., one instance of XVL on top of a radio hypervisor, serving multiple clients, i.e., the different tenants. The clients can send messages to the server for querying information about the use of radio resources or requesting radio resources at any time. Fig. 3.2 depicts the process of negotiating and using radio resources through XVL. The clients can request the allocation of separate virtual radio receivers and transmitters, each of which can have different bandwidths and can operate in different centre frequencies. Currently, XVL only ensures that the requested virtual radio resources fall within the underlying physical radio hardware's operating bandwidth, and do not overlap with the resources already allocated to other virtual radios. However, MNOs are free to set limits on the radio resource allocation per tenant, or attribute a cost to it. Upon successful negotiation and reservation of radio resources, XVL interfaces with the underlying radio hypervisor for creating or destroying the virtual radios. XVL has callbacks for informing the radio hypervisor about changes in the resource mapping or allocation of virtual radios, so the radio hypervisor can instantiate a new virtual RF front-end. Once the radio hypervisor completes the creation of the virtual RF front-end, XVL performs additional configurations in the virtual radio (to be further discussed in Section 3.2.2). Finally, XVL informs the client of the proper ways to reach its virtual radio, e.g., through CPRI, OBSAI, or a UDP socket, from which the clients can start transmitting/receiving streams of IQ samples.

In Appendix A, we detail XVL's interfaces for managing virtual radios, including both its northbound interface for tenants to negotiate and query the status of their virtual radio resource allocation, and its southbound interface for communicating with a technology-agnostic radio hypervisor, e,g., HyDRA.

At present, XVL supports radio virtualisation in the frequency domain. However, XVL separates the resource management from the internal processes employed by the radio hypervisor, which allows the resource manager to be extended and modified with ease. Thus, XVL can easily be extended for supporting time or space instead of frequency resources, or even incorporating time and space as new degrees of freedom in the radio resource definition. Ultimately, it only depends on the radio virtualisation mechanisms of the underlying radio hypervisor. Moreover, the radio resources are currently defined in terms of a centre frequency, a bandwidth, a client ID, and an UDP port.

3.2.2 Independence between Virtual Radios

Every virtual radio has its own independent virtual RF front-end, which the virtual radio employs for transmission and/or reception. Each virtual RF front-end uses a fraction of the total radio resources of the real RF front-end. The radio resources can be defined in terms of bandwidth, timeslot, or antennas, depending on the radio hypervisor's approach for virtualising the real RF front-end. Regardless of the type of radio resource, each virtual radio must send an appropriate number of IQ samples to the radio hypervisor at precise timing (to be detailed in Section 3.3). The radio hypervisor consumes the IQ samples of all virtual radios at once, multiplexing them into a single stream of IQ samples. The number of necessary IQ samples per virtual radio varies according to the amount of radio resources per virtual RF front-end. In the case of an FFT-based hypervisor, e.g., HyDRA, the bandwidth of each virtual RF front-end is mapped onto a number of FFT bins, and the virtual radios must generate a number of samples equal to the number of FFT bins to create an FFT window. Then, the radio hypervisor multiplexes the windows of the virtual RF front-ends together using an IFFT [21].

XVL employs timed buffers to ensure that the radio hypervisor receives the appropriate number of IQ samples at the right moment. Based on the sampling rates of the virtual radio and the radio hypervisor, the timed buffers output IQ samples at the precise moment for the radio hypervisor's consumption (we will present more details about this procedure in Section 3.3). Fig. 3.3 illustrates the operation of timed buffers inside a virtual radio, where UDP sockets are used for receiving/sending IQ samples from/to tenants. In the case of overflows, XVL's internal timed buffer will start filling, until reaching a cap. After that, incoming samples are dropped to prevent the virtual radio process

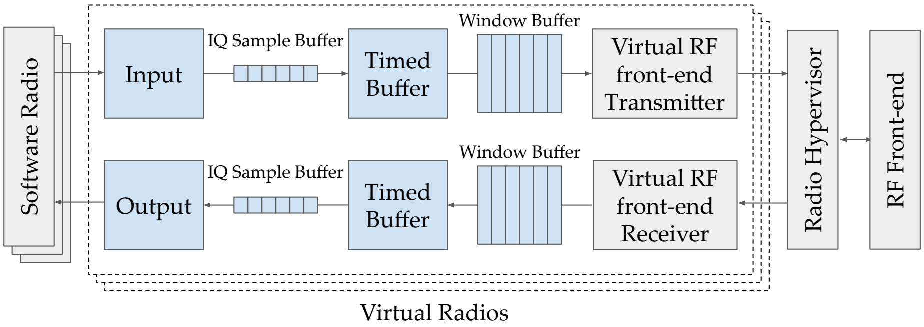


FIGURE 3.3: Timed buffers consume the received IQ samples and generate the windows that the radio hypervisor requires. The reverse process occurs in the opposite communication direction.

from overflowing memory. In the case of underflows, XVL pads the streams of IQ samples of empty windows, i.e., fills them with zeroes, while waiting for the missing samples. In this way, the radio hypervisor and the remainder of the system will not halt waiting for the delayed or missing IQ samples of a given virtual radio.

The virtual radios can either operate as transmitters, receivers, or transceivers. However, the virtual radios may not be symmetrical in both communication directions, i.e., they may employ different RATs, numerologies, or require different amounts of radio resources. For that reason, every virtual radio in XVL possesses completely independent transmitter and receiver chains. Fig. 3.3 shows an example of a virtual radio with both types of chains: the transmit chain (top) receives samples from a software radio, and the receive chain (bottom) sends samples towards a software radio. The radio hypervisor may realise each type of chain in different manners, i.e., different virtualisation approaches, or use different physical radio hardware transmission and reception.

3.2.3 Architecture and Monitoring Capabilities

Fig. 3.4 illustrates the overall architecture of the XVL implementation. The clients can interface with the server for resource discovery and negotiation through ZMQ messages [85]. The server queries the resource manager and tries to fulfil requests. The resource manager has a list of virtual radios and can interact with the radio hypervisor for creating or destroying virtual RF front-ends. Each virtual radio has an Input/Output (I/O) interface, timed buffers and a virtual RF front-end, which is tied to the radio hypervisor. Aside from the functionality for supporting the instantiation of virtual radios on-demand, XVL also has a monitor utility for inspecting the KPIs of the virtual radios.

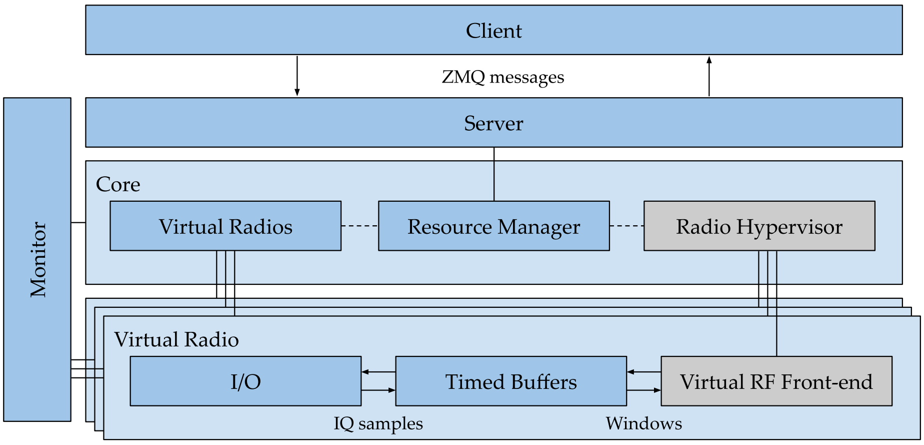


FIGURE 3.4: Block diagram showing the architecture of XVL, the elements that compose it (blue), and the elements it must interface with (grey).

Currently, the monitoring utility is used for assessing the use of computational resources per virtual radio. It measures the buffer sizes, the sampling rates, and the time each virtual radio has to fill multiple windows. Based on the buffer sizes, it is possible to determine whether the software radio is presenting overflows, i.e., buffer sizes increasing steadily, or underflows, i.e., buffer sizes frozen at zero. We plan to extend this mechanism to evaluate the introduced SINR degradation and delay. For further information, the reader can refer to the XVL's public repository (`https://bitbucket.org/joaofelipesantos/xvl/src/master/`).

3.3 Radio Processing Isolation

In this section, we focus on formalising and addressing the radio processing isolation functionality. We describe the reasoning behind the operation of the timed buffers, and derive the necessary amount of padding for isolating the processing of the streams of IQ samples of the virtual radios.

The radio virtualisation mechanisms of technology-agnostic radio hypervisors are multiplexing operations, where each virtual radio has access to a fraction of the overall radio resources of the physical radio. The existing technology-agnostic radio hypervisors (as shown in Table 3.1) employ frequency-domain radio virtualisation mechanisms, which partition the bandwidth of a physical radio's RF front-end (B) into spectrum chunks, abstracted to the virtual radios in the form of the virtual bandwidth (b_i) of their virtual RF front-end, $i = 1, \ldots, N$. The virtual radios interact with their virtual RF front-end as they would with a real RF front-end, sending/receiving streams of IQ samples. The radio virtualisation mechanism ensures isolation between the virtual radios at the radio resource level, using non-overlapping radio resources and including the necessary guard bands and/or intervals.

The multiplexing of the IQ samples generated by multiple virtual radios occurs on a per-window basis, where the radio hypervisor collects a certain number of IQ samples (M) from/to the virtual radios over a time period (τ_M) of duration equal to the ratio between M and the sampling rate of the physical radio (f_s), i.e., $\tau_M = M/f_s$. Only when in possession of M IQ samples, the radio hypervisor is able to multiplex/demultiplex the IQ samples from/to multiple virtual radios. For frequency-domain radio virtualisation mechanisms, each virtual radio is expected to produce/consume a given number of IQ samples (m_i) within τ_M, corresponding to a fraction of M equal to the ratio between its virtual bandwidth b_i and the total bandwidth B:

$$m_i = \left\lceil M \cdot \frac{b_i}{B} \right\rceil . \tag{3.1}$$

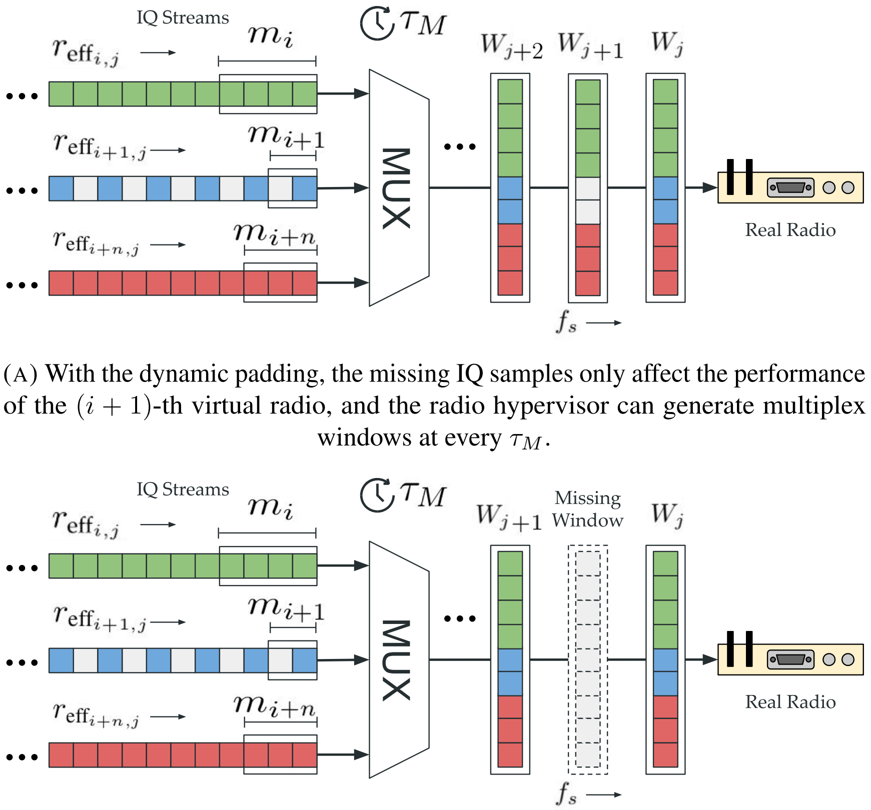
The ceiling operation ($\lceil . \rceil$), required to ensure an integer number of m_i samples, may introduce a mismatch between the allocated and requested virtual bandwidth, leading to signal distortions [74]. For simplicity, we restrict the choice of b_i such that m_i is always an integer, avoiding this mismatch.

On a RANaaS platform, the virtual radios send/receive streams of IQ samples to/from different tenants. It is likely that the virtual radios will not always produce/consume the exact number of m_i IQ samples during every multiplex window, i.e., generate samples at an expected nominal rate $r_i = m_i/\tau_M$. Instead, during the j-th multiplex window (W_j), the i-th virtual radio will present an effective rate ($r_{\text{eff}_{i,j}}$), which may vary for each window and can be inferior to the expected nominal rate ($r_{\text{eff}_{i,j}} \leq r_i$). Such an effective rate may be the result of: (*i*) the multiple access scheme of RATs which intermittently transmit frames, e.g., WiFi and Bluetooth; (*ii*) the behaviour of software radios and BBUs which cannot process IQ samples at the expected nominal rate; or (*iii*) RAN slices that stopped operating but were not yet de-allocated from the RANaaS platform.

To prevent individual virtual radios from affecting the processing performance of the entire system, we must isolate the processing of their streams of IQ samples and ensure that the appropriate number of M IQ samples is available to the radio hypervisor at every τ_M. We can achieve such isolation through dynamic padding of the streams of IQ samples, i.e., with the inclusion of a number of null samples ($v_{i,j}$) to the stream of the i-th virtual radio during W_j whenever $r_{\text{eff}_{i,j}} < r_i$ so as to provide a total of M samples to fill W_j:

$$v_{i,j} = m_i - \lfloor r_{\text{eff}_{i,j}} \cdot \tau_M \rfloor \quad = m_i - \left\lfloor \frac{m_i \cdot r_{\text{eff}_{i,j}}}{r_i} \right\rfloor = \left\lceil m_i \cdot \frac{r_i - r_{\text{eff}_{i,j}}}{r_i} \right\rceil . \tag{3.2}$$

The dynamic padding limits the effect of a virtual radio presenting a $r_{\text{eff}_{i,j}} < r_i$ to its own RAN slice, without delaying the production/consumption of multiplex windows, as shown in Fig. 3.5a. If such padding was not added, then the radio hypervisor would need to wait until a full window of M IQ samples was available, as shown in Fig. 3.5b. This implies that the radio hypervisor would not be able to ensure independence between virtual radios in terms of the processing of their streams of IQ samples, i.e., the virtual radios would not be isolated at the radio processing level.



(A) With the dynamic padding, the missing IQ samples only affect the performance of the $(i + 1)$-th virtual radio, and the radio hypervisor can generate multiplex windows at every τ_M.

(B) Without the dynamic padding, the missing IQ samples affect all virtual radios, as the radio hypervisor cannot generate windows at every τ_M and has to wait until it collected M IQ samples.

FIGURE 3.5: Multiplexing of multiple virtual radios with and without the dynamic padding of IQ samples, when the $(i + 1)$-th virtual radio presents an effective rate inferior to its expected nominal rate. The use of dynamic padding ensures isolation between virtual radios at the level of processing their IQ samples, preventing any virtual radio from delaying the production/consumption of multiplex windows.

The impact of this delay would accumulate over successive windows, deteriorating the performance experienced by all virtual radios over time. Based on the number of IQ samples collected for generating M_j during a τ_M (C_j), the multiplex window index (j), and f_s, we can calculate the experienced accumulated delay (τ_{exp}) for producing/consuming the given window:

$$\tau_{\text{exp}} = \sum_{k=1}^{j} \frac{k}{f_s} \cdot (M - C_k)\,. \tag{3.3}$$

The τ_{exp} expresses the total delay experienced by each tenant during the operation of their RAN slices if any of the virtual radios present a $r_{\text{eff}_{i,j}} < r_i$. In Fig. 3.6, we illustrate the impact of τ_{exp}. We show a numerical analysis in which we consider a constant $r_{\text{eff}_{i,j}}$ for every window. It demonstrates how τ_{exp} mounts over successive windows and would degrade the operation of the RANaaS platform over time, if no padding were applied.

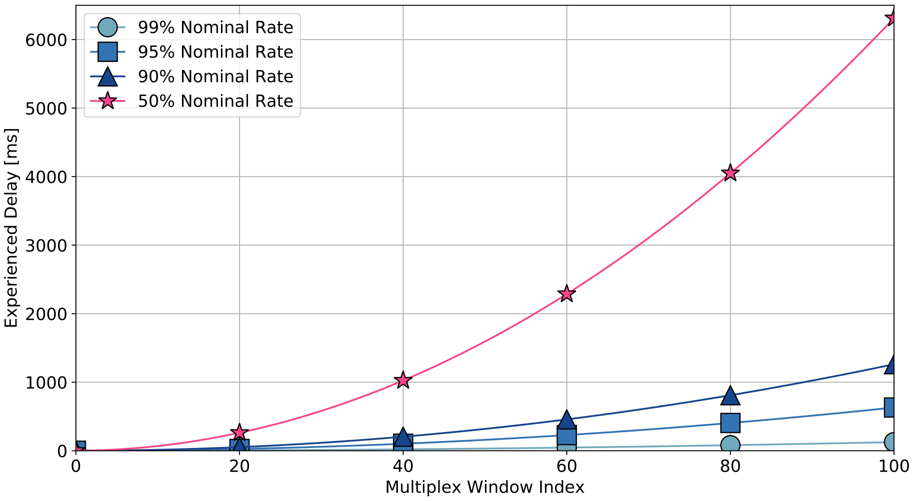


FIGURE 3.6: How different effective rates would affect the experienced delay over time, if dynamic padding were not applied. In this scenario, we have a radio hypervisor at $f_s = 200$ KHz and $M = 1000$ samples, with two virtual radios at $r_i = 100$ KHz and $m_i = 500$ samples, while one of the virtual radios manifests an effective rate shown as a percentage of the expected nominal rate.

3.4 Experimental Evaluation

In this section, we validate the use of XVL for leveraging radio virtualisation to support RANaaS. First, we describe the integration between XVL and HyDRA, forming a RANaaS platform capable of deploying heterogeneous RAN slices as a service. Then, we assess the performance of our RANaaS platform regarding the computational overhead, the delay in provisioning virtual radios, the delay introduced by radio virtualisation, and the signal degradation due to the virtualisation mechanism.

3.4.1 Integration with HyDRA and GNU Radio

We integrated XVL with HyDRA [21] for validating XVL's operation and showcasing its features. This integration enables HyDRA to support RANaaS, through the provision of tailored virtual radios on-demand. In addition, we developed a client-side library for communicating with XVL. This library allows any C++-based client to interface with XVL for querying, requesting, and using XVL's radio resources. We employ ZMQ, a cross-platform networking library, which facilitates porting our communication library to serve clients implemented in different programming languages, e.g., Python, Java, or Ruby.

In order to facilitate the use of our contribution by the research community, we have also integrated XVL with GNU Radio, a widely known SDK for prototyping and developing on SDR platforms. GNU Radio realises RATs and signal-processing systems as acyclic directional graphs, known as flowgraphs, which represent continuous streams of IQ samples between signal processing blocks, e.g., modulation, coding, and filtering. Each flowgraph possesses one or more source blocks for inserting IQ samples into the flowgraph, and one or more sink blocks for exporting IQ samples from the flowgraph. We used the ZMQ library for developing GNU Radio blocks that can seamlessly replace the regular USRP source and sink blocks in existing GNU Radio flowgraphs. These new blocks only require the extra information of the IP address and port number of where an XVL instance is running, as well as a client ID to identify the tenant. The benefits of this setup are twofold: it allows us to use an existing radio hypervisor and GNU Radio flowgraphs for evaluating XVL, and it enables current GNU Radio flowgraphs to leverage XVL for instantiating virtual radios.

3.4.2 Computational Overhead

In this analysis, we consider the computational overhead introduced by running XVL alongside a radio hypervisor to serve an LTE and an NB-IoT virtual radios. XVL and the radio hypervisor introduce a new layer of complexity between the software radios and the RF front-end. We measured the CPU utilisation for an XVL and HyDRA instance for different IFFT sizes and normalised it as a percentage of one core in an Intel Xeon E52620 v2 processor. Fig. 3.7 shows the results of our measurements. Independently of the configuration used in the radio hypervisor, the combination of XVL and HyDRA, the NB-IoT, and the LTE, require roughly 60%, 5% and 30% of the CPU processing power, respectively. The constant CPU utilisation by the virtual radios is expected, as their processing is independent of the radio hypervisor. However, we did not capture any increase in the CPU overhead for the radio hypervisor as reported in [21]. We attribute this to the multithreading that XVL employs for managing the timed buffers and sockets, decreasing the significance of the CPU utilisation of the radio hypervisor. These results indicate that XVL can run alongside a radio hypervisor on COTS computing hardware for deploying multiple virtual radios.

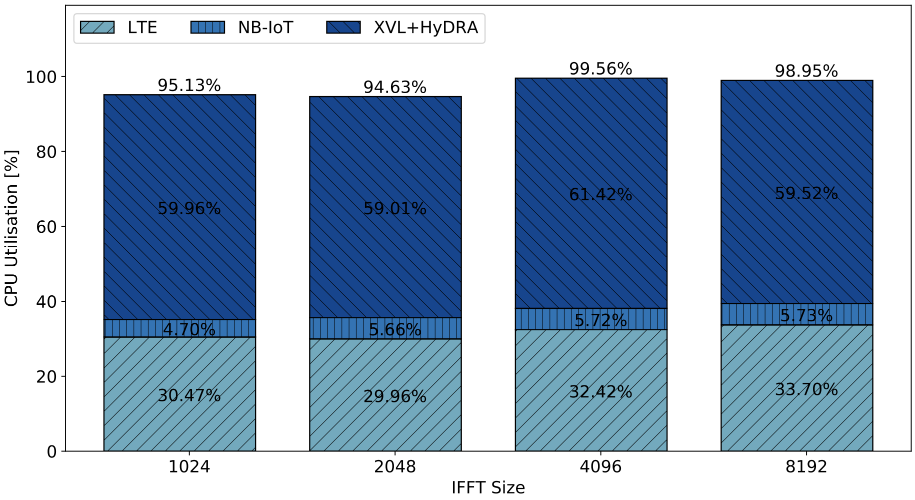


FIGURE 3.7: CPU utilisation when running XVL alongside HyDRA, in addition to an LTE and an NB-IoT virtual radios, under different radio hypervisor configurations.

3.4.3 Provisioning Delay

In this analysis, we are interested in the delay for provisioning virtual radios with XVL. The provisioning delay is crucial for planning and evaluating a RAN deployment using the RANaaS paradigm, as it dictates the time interval required to provision the virtual radios that realise a RAN slice. This interval comprehends the time between the client sending a radio resource request, and XVL returning the resource allocation confirmation (there are several procedures that occur between these two events, as we have seen in Section 3.2). Fig. 3.8 shows the results of our measurements for different configurations of the virtual radio and radio hypervisor. The exponential growth in the semi-log scale denotes a linear behaviour in relation to the number of radio resources allocated per virtual radio, regardless of the IFFT size of the radio hypervisor. We attribute the linear behaviour to the time for allocating memory for XVL's buffers. The worst case scenario, where a virtual radio required 8192 FFT bins, took less than 300 ms to be fulfilled, hence, indicating that XVL is capable of provisioning virtual radios on the fly.

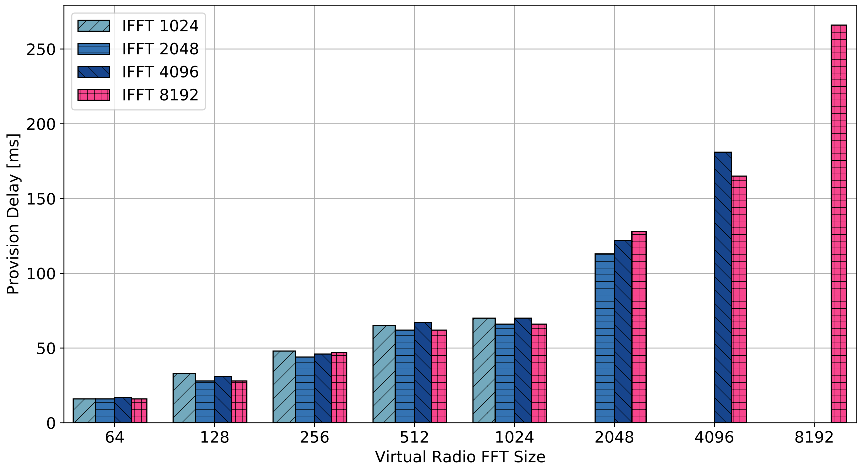
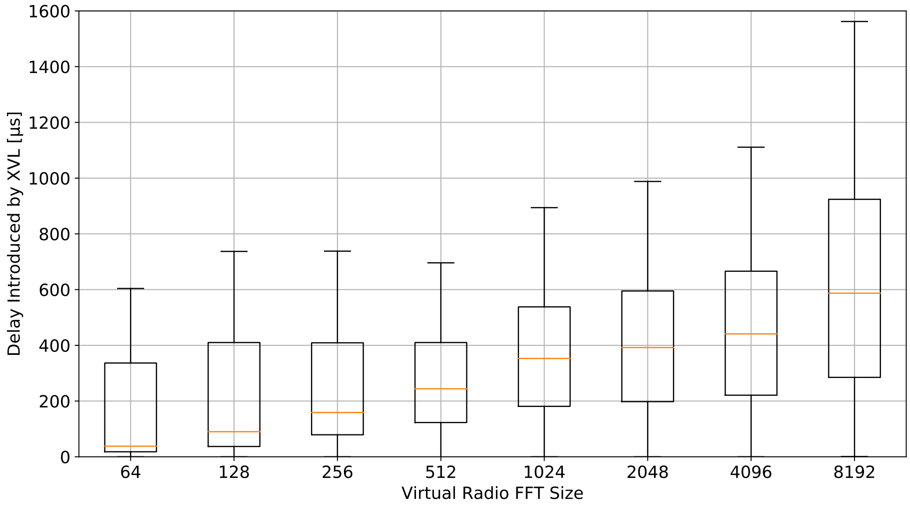


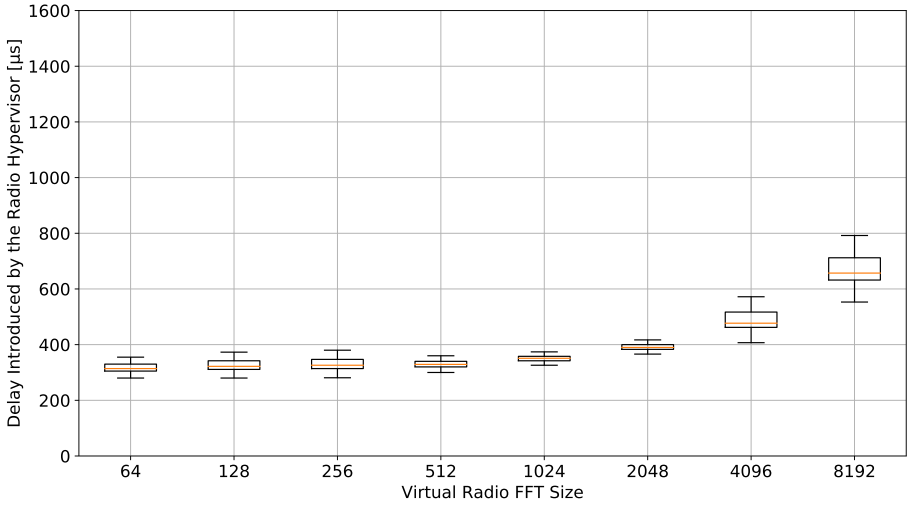
FIGURE 3.8: The provisioning delay to fulfil a virtual radio request. The time taken for creating virtual radios is proportional to the number of allocated FFT bins, but not to the overall IFFT size of the radio hypervisor.

3.4.4 Service Delay

In this analysis, we are interested in the delay that XVL and HyDRA introduce to serve communication services. The service delay is crucial to analyse the impact of relying on virtualisation to realise communication services (using virtual instead of real radio resources), and to evaluate whether it can impair their operation. First, we measured the time spent by XVL for delivering the IQ samples to the radio hypervisor, i.e., from successful UDP receptions, through the timed buffers, and the generation of a multiplex window. Fig. 3.9a shows the results of our measurements, with a median of 343 μs. Then, we measured the time spent by HyDRA to virtualise the RF front-end, i.e., consume the window, multiplex the IQ samples of all virtual RF front-ends, and forward the IQ samples to the real RF front-end. Fig. 3.9b shows the results of our measurements, with a median of 334 μs. These results show that XVL introduces a service delay within the same order of magnitude as the radio hypervisor. The median delay introduced by using both XVL and HyDRA to support RANaaS remains under 1 ms, but it still can be impactful in the delay budget for C-RAN scenarios or for the MAC scheme of RATs with stringent timing requirements, and hence, must be taken into account when planning to realise a communication service using RAN slices. In addition, such service delay can also prevent the use of our solution in real-world 5G NR deployments, where each virtual radio may require up to 100 MHz of bandwidth. However, it is worth mentioning that both HyDRA and XVL were implemented as proof-of-concept prototypes, focusing on validating their respective contributions, instead of focusing on delivering the best performance. We believe that more efficient implementations of both HyDRA and XVL could reduce the service delay to a fraction of the current value. For example, one could leverage the USRP's embedded FPGA to implement part of the complex and resource-intensive DSP operations, e.g., the series of FFTs and IFFTs, which can reduce the delay introduced by this part of the signal processing pipeline by a factor of 15 [86].



(A) The service delay introduced by XVL. The service delay increases with the number of radio resources, due to the longer time required to allocate larger windows in memory.



(B) The service delay introduced by HyDRA. The service delay increases with the number of radio resources, due to the increasing complexity of the FFT/IFFT operations.

FIGURE 3.9: Delay measurements for different virtual radio configurations. All the measurements were made using a real bandwidth of 2 MHz and an overall IFFT size of 8192 bins, which corresponds to the worst case scenario in Section 3.4.3.

3.4.5 Signal Degradation

In this analysis, we consider how the resource allocation granularity of the radio hypervisor affects on the performance of virtual radios. FFT-based radio hypervisors, e.g., HyDRA, employ IFFTs to partition the bandwidth of a physical radio (B) into a number of FFT bins equal to the IFFT size (n), each with a bin width $\Delta f = B/n$. The i-th virtual radio is provided with an integer number of FFT bins (n_i) of total bandwidth ($\tilde{b_i}$) that approximates its requested bandwidth (b_i), according to $n_i = \lceil b_i/\Delta f \rceil$. Therefore, for a fixed B, the IFFT size dictates the resource allocation granularity of the radio hypervisor, i.e., the minimum amount of radio resources which the radio hypervisor can allocate to a virtual radio, as shown in Fig. 3.10. The value of n not only affects the efficiency of the resource allocation of the radio hypervisor, i.e., influencing the overprovisioning of real radio resources used to create virtual radios, but it also affects the quality of the radio virtualisation mechanism. A mismatch between b_i and $\tilde{b_i}$ leads to a decoupling of the sampling rates expected by the software radio and produced/consumed by the virtual RF front-end, causing signal distortions that degrade the waveform of the virtual radios proportional to the bandwidth mismatch [74] [87]. We measured how different IFFT sizes, and consequently, resource allocation granularities, affect the performance of the virtual radios in terms of the Mean Square Error (MSE), and Table 3.2 shows the result of our evaluation. These results show how the performance impact of radio virtualisation on different RATs varies with the resource allocation granularity of the radio hypervisor. Moreover, at an IFFT size of 8192, we were able to achieve a performance comparable to RATs running on bare metal, i.e., without radio virtualisation.

3.5 Conclusions

In this chapter, we addressed the resource management functionality for enabling RANaaS through radio virtualisation. We compiled a comprehensive list of features that radio hypervisors must incorporate for supporting RANaaS. We then evaluated the state-of-the-art on radio virtualisation with respect to these requirements and identified what we consider the most suitable radio hypervisor for developing a prototype RANaaS platform. Next, we investigated the challenges for ensuring isolation between virtual radios at the radio processing level, and proposed dynamic padding to maintain multiplex synchronicity. We presented XVL, a software layer that can be added on top of existing hypervisors and provide them with the missing capabilities for supporting the RANaaS paradigm, namely: (*i*) a radio resource broker for negotiating radios resources; (*ii*) timed buffers for dynamically padding the streams of IQ samples; and (*iii*) a monitoring interface for tenants to query the performance of their RAN slices. We evaluated XVL regarding its computational overhead, provisioning delay, and service delay. Our results show that XVL introduces a delay comparable to a radio hypervisor, can run on COTS computing hardware, and provision virtual radios in real-time. Currently, the main limitations of our solution are: (*i*) the present service delay, which we believe can be severely decreased with more efficient implementation and the use of hardware acceleration; and (*ii*) the limited availability and potentially prohibitive cost of SDRs with ultra-wide bandwidth, which should become more accessible as the demand for broader bandwidth RF front-end increases due to economy of scale. Nonetheless, our findings confirm that the combination of XVL with a technology-agnostic radio hypervisor such as HyDRA can act as a RANaaS platform capable of deploying heterogeneous RAN slices as a service, enabling infrastructure sharing and potential cost reductions for MNOs serving special-purpose RATs or operating in CBRS and U-NII bands.

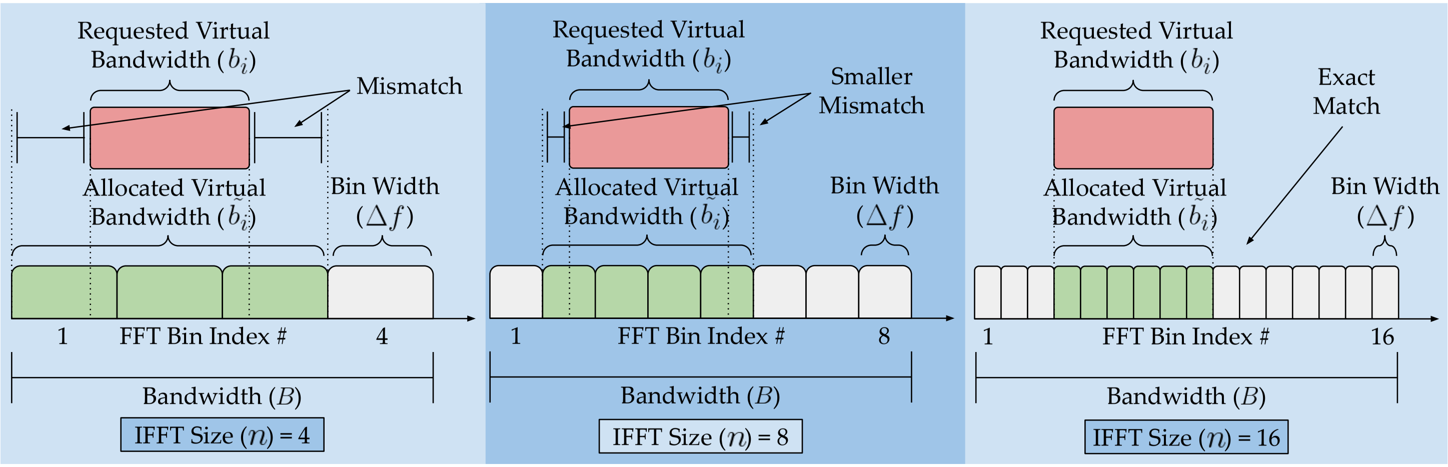


FIGURE 3.10: An example of the resource allocation procedure of FFT-based radio hypervisors. The virtual radios request a certain bandwidth (b_i) at a given centre frequency, and are provided with an allocated virtual bandwidth ($\tilde{b_i}$) comprised of an integer number of contiguous FFT bins (n_i). For a fixed B, the IFFT size (n) determines the FFT bin width (Δf), and hence, the radio hypervisr's resource allocation granularity. These values can potentially be tweaked to reduce mismatches in the resource allocation, to both save real radio resources and reduce signal degradations.

		LTE			NB-IoT		
	Granularity (Δf)	FFT Size (n_i)	Bandwidth ($\tilde{b_i}$)	MSE	FFT Size (n_i)	Bandwidth ($\tilde{b_i}$)	MSE
Bare metal	–	–	1.400,0 KHz	2.1 dB	–	200,0 KHz	1.1 dB
IFFT 8192	977 Hz	1433	1.400,2 KHz	2.4 dB	205	200,2 KHz	1.1 dB
IFFT 4096	1953 Hz	717	1.400,3 KHz	3.4 dB	103	201,2 KHz	2.2 dB
IFFT 2048	3906 Hz	359	1.402,3 KHz	6.1 dB	52	203,3 KHz	4.1 dB
IFFT 1024	7812 Hz	180	1.406,2 KHz	11.2 dB	27	210,9 KHz	7.9 dB

TABLE 3.2: A comparison between the MSE for over-the-air transmissions using the RATs discussed in Section 3.4.2, running both on bare metal and on top of HyDRA with a fixed B of 8 MHz and different IFFT sizes.

Chapter 4

Embedding Heterogeneous RAN Slices on RANaaS Platforms

Embedding Heterogeneous RAN Slices on RANaaS Platforms

> *Your focus determines your reality.*
>
> Qui-Gon Jinn, Star Wars: Episode I – The Phantom Menace, *1999*

In this chapter, we address the embedding of heterogeneous RAN slices for technology-agnostic RANaaS. We propose a novel formulation using a Travelling Salesman Problem (TSP)-based approach for mapping real radio resources to realise virtual radios that is: (*i*) technology-agnostic, leveraging radio slicing in the frequency domain to support RAN slices with different RATs and numerologies; (*ii*) extensible, supporting new types of RATs based only on their requirements for isolation in the frequency domain; and (*iii*) secure, separating RAN slices down to the PHY layer. To the best of our knowledge, these are the first model and formulation developed for embedding heterogeneous RAN slices, leveraging technology-agnostic radio hypervisors. In addition, we introduce a resource management optimisation problem to be solved by the MNOs, to determine the optimal arrangement of RAN slices for minimising the total isolation overhead on their RANaaS platforms. Due to this problem's Mixed-Integer Linear Programming (MILP) nature, we also present two heuristic algorithms and compare their performance for solving the problem against the optimal solution found using a linear programming solver.

The technical work presented in this chapter is based on our work "Optimal Embedding of Heterogeneous RAN Slices for Technology-agnostic RANaaS" submitted to IEEE Wireless Communication Letters [88].

The remainder of this chapter is organised as follows. In Section 4.1, we briefly review the state-of-the-art on models and solutions for embedding virtual wireless networks. In Section 4.2, we introduce our novel TSP-based formulation for embedding heterogeneous RAN slices; propose a resource management optimisation problem for minimising the total isolation overhead; and present two heuristic algorithms as alternatives for solving this problem using a lower computational footprint. In Section 4.3, we numerically evaluate the performance and computational footprint for running the different solutions. Finally, in Section 4.4, we conclude this chapter and pose our final remarks.

4.1 Current Approaches for Embedding RAN Slices

As discussed in Section 2.2, most experimental research efforts on RAN slicing adopt technology-specific radio hypervisors, leveraging the resource allocation capabilities of certain RATs for creating RAN slices [22]. In particular, the vast majority of existing radio hypervisors employ grid-based radio virtualisation mechanisms for slicing LTE or WiMax deployments. Due to their popularity, the current literature on the embedding of RAN slices focuses on proposing optimal resource management solutions for grid-based radio hypervisors. These works model the embedding of RAN slices as variations of the bin packing problem, placing virtual radios in a regular radio resource grid as illustrated in Fig. 4.1, and propose different methods to solve these resource management problems, e.g., using auctions [83] [89], Karnaugh maps [90] [91], or machine learning [92] [93].

The modelling approach mentioned above and the associated solutions assume that the underlying physical radio platform and every RAN slice possess the same type and granularity of radio resources, which does not hold true for the virtual radios with different RATs or numerologies in technology-agnostic RANaaS platforms. In addition, any viable solution for embedding heterogeneous RAN slices must factor in the resource allocation overhead for including guard bands between virtual radios to avoid interference from OOBE [94]. The amount of overhead can range from one subcarrier to a few Megahertz depending on the RAN slices' waveforms, subcarrier spacings, tolerance to Signal-to-Interference Ratio (SIR), among other parameters [95] [96]. However, any extensions to current Knapsack-based formulations for considering conditional guard bands between each pair of RAN slices would greatly increase the problem's complexity [97], and consequently, the time required for solving it (to be discussed in the next section). Therefore, we observe that the current models and solutions are not suitable for embedding heterogeneous RAN slices.

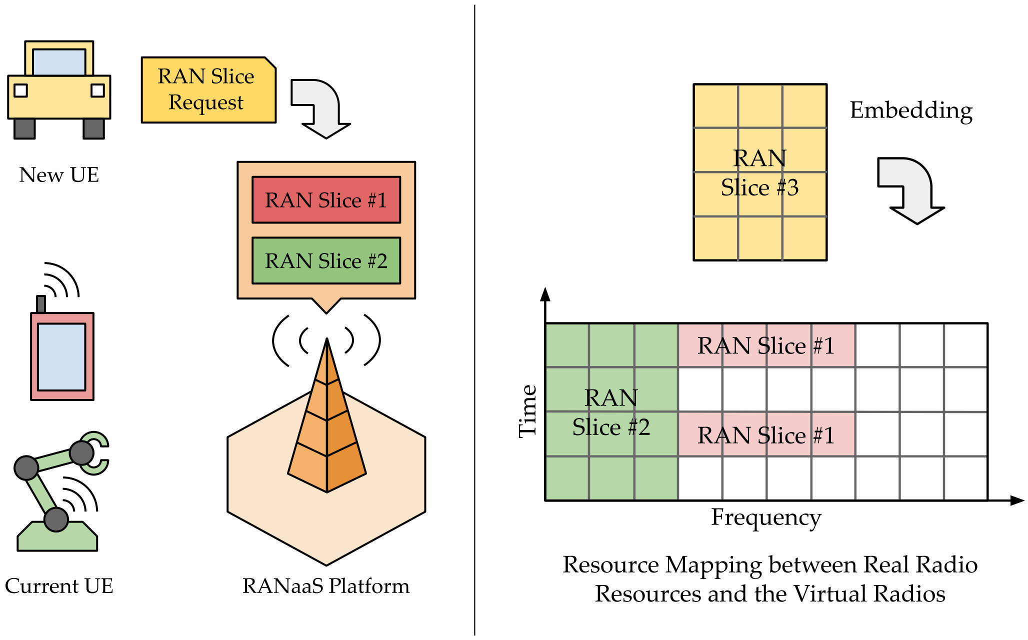


FIGURE 4.1: An example of a RANaaS platform fulfilling a new RAN slice request (left). The radio hypervisor must determine the most efficient mapping of radio resources from the underlying physical radio hardware to realise the virtual radios with different requirements (right).

4.2 Embedding Heterogeneous RAN slices

In this section, we start by demonstrating how current Knapsack-based virtual wireless network embedding approaches be cannot efficiently extended to consider the additional overhead required to ensure isolation between heterogeneous RAN slices. Then, we introduce a novel model and solution for embedding heterogeneous RAN slices, leveraging their characteristics and isolation requirements by design. We propose a graph representation for modelling RAN slices on RANaaS platforms. Next, we introduce a Freely Open-loop TSP (FOTSP) resource management optimisation problem that finds the optimal embedding of RAN slices for minimising the total isolation overhead, i.e., the total amount of radio resources consumed with guard bands. Finally, we detail two heuristic algorithms for solving the embedding problem in using a lower computational footprint.

4.2.1 Extending Knapsack Formulations to Support Heterogeneous RAN Slices

Let us consider the Knapsack-based modelling detailed in [98]. The formulation presented in this work can be readily employed in the context of virtual wireless network embedding, being generic enough to consider any number of dimensions of the radio resources, e.g., time, frequency, and space. Without loss of generality, here we focus on the one-dimensional case for demonstrating the extension of Knapsack-based models to support heterogeneous RAN slices, which is relative to a regular radio resource grid of frequency resources and can be seen as a particular case of grid-based radio hypervisors. Akin to the vast majority of works in virtual wireless network embedding using a Knapsack-based modelling [90], [91], [99]–[102], the main goal of the problem presented in [98] is to maximise the total revenue given by embedding a subset of the existing n RAN slices using the available radio resources. This formulation assumes a constant profit value p_i relative to embedding the i-th RAN slice, and uses a binary indicator variable x_i, equal to 1 if RAN slice i is embedded in the RANaaS platform, and 0 otherwise. For the one-dimensional case, the frequency coordinate of RAN slice i is f_i, meaning that the lowest frequency of the RAN slice's virtual bandwidth (b_i) will be located at this frequency value. If a RAN slice is not embedded within the RANaaS platform, we can assume that f_i = 0. As the total bandwidth occupied by the RAN slices has to fit within the real bandwidth of the RF front-end (B), we have that:

$$0 \leq f_i \leq B - b_i. \tag{4.1}$$

We specifically chose the formulation in [98] for demonstrating the complexity of extending Knapsack-based models to consider the isolation overhead between heterogeneous RAN slices because it captures the sense of adjacency, i.e., the relative placement of RAN slices embedded next to each other, which we will need for determining the minimum required guard bands between each pair of RAN slices. It does so by introducing binary decision variables, $l_{ij,}$ (left) and $r_{i,j}$ (right), to indicate the relative position of RAN slices i, j where $i < j$. To ensure that no two RAN slices i, j overlap, it requires that:

$$l_{i,j} + r_{i,j} = 1, \tag{4.2}$$

whenever RAN slices i and j are both embedded, i.e., $s_i = s_j = 1$. Depending on the relative position of two RAN slices, their frequency coordinates must satisfy the following inequalities:

$$\begin{aligned} l_{i,j} = 1 \Rightarrow f_i + b_i \leq f_j \\ r_{i,j} = 1 \Rightarrow f_j + b_j \leq f_i. \end{aligned} \tag{4.3}$$

The one-dimensional Knapsack problem for embedding RAN slices can be formulated as follows:

$$
\begin{aligned}
P1 : \max_{x_i} \quad & \sum_{i=1}^{n} p_i \cdot x_i && \text{(4.4a)}\\
\text{s.t.} \quad & l_{i,j} + r_{i,j} = x_i + x_j - 1, && \text{(4.4b)}\\
& f_i - f_j + B \cdot l_{i,j} \leq B - b_i, && \text{(4.4c)}\\
& f_j - f_i + B \cdot r_{i,j} \leq B - b_j, && \text{(4.4d)}\\
& 0 \leq f_i \leq B - f_i, && \text{(4.4e)}\\
& l_{i,j}, r_{i,j} \in \{0,1\}, && \text{(4.4f)}\\
& x_i \in \{0,1\}, && \text{(4.4g)}\\
& f_i \geq 0 && \text{(4.4h)}
\end{aligned}
$$

where Eq. (4.4a) defines the objective function that maximises the total profit gained for allocating a subset of the n existing RAN slices onto the RANaaS platform; Eq. (4.4b) ensures that if RAN slices i and j are embedded, they must be located either left or right of each other, as stated in (4.2); Eqs. (4.4c) and (4.4d) are linear versions of constraints (4.3); and Eq. (4.4e) corresponds to constraint (4.1). The solution to $P1$ yields the optimal embedding of the subset of RAN slices that maximises the profit gained by MNOs using the available real radio resources.

Now, to extend $P1$ to support the embedding of heterogeneous RAN slices, we need to modify the problem formulation to ensure isolation between RAN slices in the frequency domain. Let us assume that each pair of RAN slices requires a minimum guard band ($g_{i,j}$) between each other, an added overhead in frequency that ensures they do not interfere with one another in case they are embedded adjacent to each other. We can incorporate the conditional $g_{i,j}$ in $P1$ by modifying Eqs. (4.4c) and (4.4d) to include $g_{i,j}$ if and only if two RAN slices i, j are adjacent to one another as follows:

$$
\begin{gathered}
f_i - f_j + B \cdot l_{i,j} \leq B - (b_i + g_{i,j} \cdot l_{i,j}) \\
\therefore \\
f_i - f_j + l_{i,j} \cdot (B + g_{i,j}) \cdot \leq B - b_i .
\end{gathered} \tag{4.5}
$$

Finally, we can leverage the reasoning behind Eq. (4.5) to extend $P1$, modifying Eqs. (4.4c) and (4.4d) accordingly, and proposing a novel Knapsack-based formulation that supports heterogeneous RAN slices and their different requirements for isolation in the frequency domain, detailed as follows:

$$
\begin{aligned}
P2 : \max_{x_i} \quad & \sum_{i=1}^{n} p_i \cdot x_i && \text{(4.6a)}\\
\text{s.t.} \quad & l_{i,j} + r_{i,j} = x_i + x_j - 1, && \text{(4.6b)}\\
& f_i - f_j + l_{i,j} \cdot (B + g_{i,j}) \leq B - b_i, && \text{(4.6c)}\\
& f_j - f_i + r_{i,j} \cdot (B + g_{j,i}) \leq B - b_j, && \text{(4.6d)}\\
& 0 \leq f_i \leq B - b_i, && \text{(4.6e)}\\
& l_{i,j}, r_{i,j} \in \{0, 1\}, && \text{(4.6f)}\\
& x_i \in \{0, 1\}, && \text{(4.6g)}\\
& f_i \geq 0. && \text{(4.6h)}
\end{aligned}
$$

The problem $P2$ is very similar to $P1$, as its solution also yields the optimal embedding of the subset of RAN slices that maximises the profit gained by MNOs using the available real radio resources, but $P2$ is more general and can support the embedding of heterogeneous RAN slices that require different amounts of isolation in the frequency domain. However, we observe that $P2$ has non-linear constraints, i.e., Eqs (4.6c) and (4.6d), that result from the combination of the binary decisions variable $l_{i,j}$ and $r_{i,j}$, and the conditional guard band $g_{i,j}$ between RAN slices i, j. These constraints are necessary to make a conditional inclusion of guard bands between RAN slices, at the cost of turning $P2$ into a non-linear optimisation problem that is very hard to obtain a direct solution. Therefore, even though one can extend a Knapsack-based model to support heterogeneous RAN slices, it becomes orders of magnitude more complex than the traditional MILP formulation. This issue makes us believe that a Knapsack-based formulation (or any of its variations) is not the ideal approach to model this class of problems, which led us to investigate a new approach for modelling the embedding of heterogeneous RAN slices, detailed in the next section.

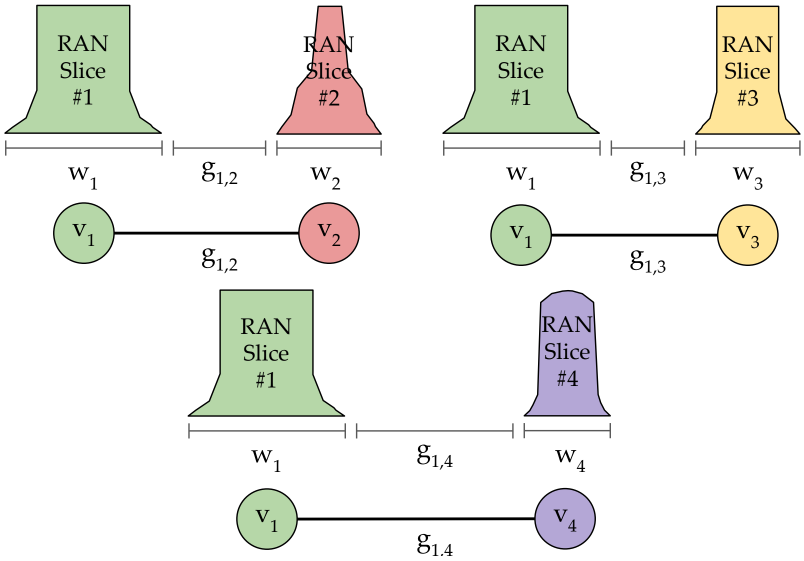


FIGURE 4.2: Distinct RATs and numerologies entail different waveforms and bandwidths (w_i), which display a varied OOBE and resilience to interference. Consequently, each pair of RAN slices requires a different separation in the frequency domain, a guard band ($g_{i,j}$), to ensure isolation from one another.

4.2.2 Graph-based Modelling of Heterogeneous RAN Slices

Let us take a new approach and consider a set of heterogeneous RAN slices on a RANaaS platform, leveraging a technology-agnostic radio hypervisor. Each RAN slice belongs to a different tenant, and is tailored with a particular RAT or numerology to serve a specific use case or application. Due to their distinct characteristics, each pair of RAN slices possesses a minimum required guard band to ensure isolation and prevent interference from the OOBEs of one another [95] [96] [103], as shown in Fig. 4.2. In this way, we can combine these pairwise relationships to represent the different heterogeneous RAN slices and their requirements for isolation in the frequency domain as a complete weighted graph $G = (V, E)$ that contains no self-edges or repeated edges. The set of nodes $V = \{v_1, \cdots, v_n\}$ represent the RAN slices; and $g_{i,j}$, the weight of edge $(i, j) \in E$, represents the minimum required guard band between v_i and v_j, as shown in Fig. 4.3. We assume that $g_{i,j} = g_{j,i}$, i.e., that the minimum required guard band between v_i and v_j is the same independent of their order of adjacency, as most RATs require the same separation in the frequency domain on the lower and upper portions of their occupied bandwidth.

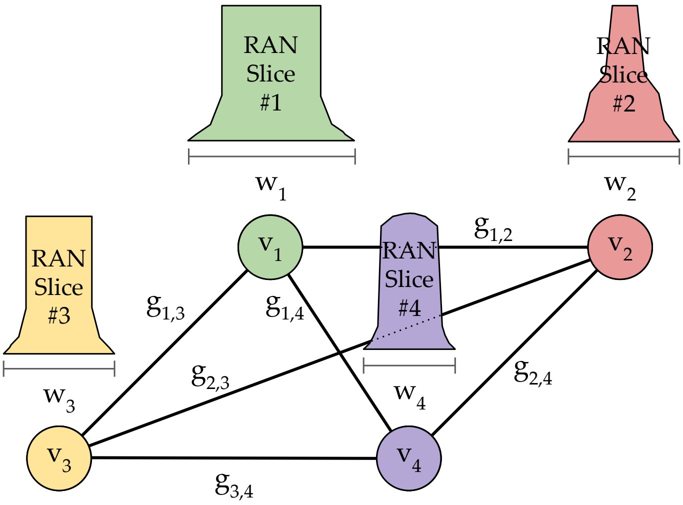


FIGURE 4.3: An illustration of the graph representation of heterogeneous RAN slices (nodes) and the required guard bands from one another (edges), forming a complete weighted graph, G.

4.2.3 Embedding RAN Slices with Minimal Isolation Overhead

Based on the complete weighted graph representation of RAN slices, we can derive the optimal solution for embedding heterogeneous RAN slices on a RANaaS platform with minimum total isolation overhead. We can achieve this by finding the path between nodes in G that has the smallest total weight, i.e., the optimal sequence of RAN slices with the smallest amount of total radio resources consumed with guard bands, as shown in Fig. 4.4. We can formulate this problem as a FOTSP, a variation of the widely known TSP problem, which for a given list of nodes and weights between each pair of nodes, searches for the path of the smallest weight that visits each node exactly once. The TSP is a combinatorial optimisation problem, important in theoretical computer science and operations research fields [104] [105].

The FOTSP variation better represents the embedding of RAN slices on the finite and linear bandwidth of a physical radio, as the allocation of RAN slices is open-loop, i.e., there are no edges (or guard bands) between the first and the last nodes (or RAN slices), nor are these two known a priori [106] [107]. The formulation of the FOTSP is as follows:

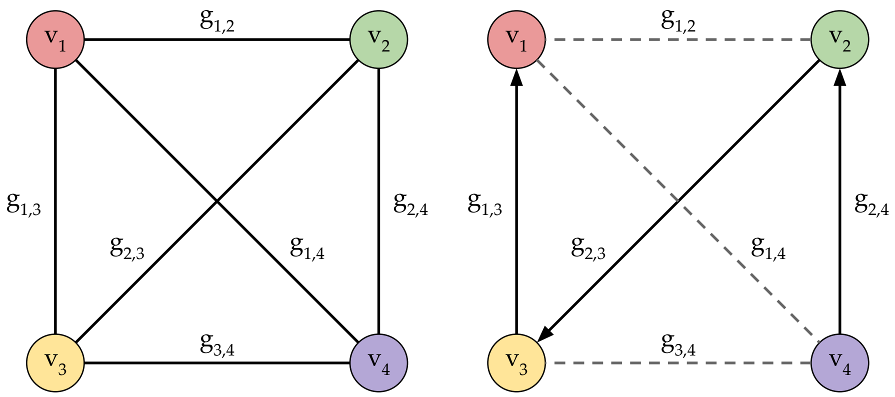


FIGURE 4.4: An example of the graph representation of RAN slices and the minimum guard band between each other (left). Based on their guard bands, we can select a path between all RAN slices that minimises the total isolation overhead (right).

$$P3 : \min_{x_{i,j}} \quad \sum_{i=1}^{n}\sum_{j=1}^{n} g_{i,j} \cdot x_{i,j} \tag{4.7a}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} x_{i,j} \leq 1, \tag{4.7b}$$

$$\sum_{j=1}^{n} x_{i,j} \leq 1, \tag{4.7c}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n} x_{i,j} = |V| - 1, \tag{4.7d}$$

$$\sum_{i\in S}\sum_{j\in S} x_{i,j} \leq |S| - 1, \quad S \subset V : S \neq \emptyset, \tag{4.7e}$$

$$x_{i,j} \in \{0,1\} \tag{4.7f}$$

where $x_{i,j}$ is a binary indicator variable which is equal to 1 if RAN slice *i* is allocated adjacent to RAN slice *j*, and 0 otherwise. Eq. (4.7a) defines the objective function that minimises the total amount of radio resources consumed with guard bands; Eqs. (4.7b) and (4.7c) are the degree constraints, ensuring that each node is connected to at most one other node, forming a path between consecutive RAN slices and preventing the formation of branches; Eq. (4.7d) is the cardinality constraint, limiting the number of edges connecting all nodes to form an open loop, i.e., an acyclic path representing the continuous bandwidth occupied by consecutive RAN slices and their guard bands; and Eqs. (4.7e) are known as the set of subtour elimination constraints [104], lazy constraints generated and added to the problem while we are solving it, that serve to prevent solutions comprised

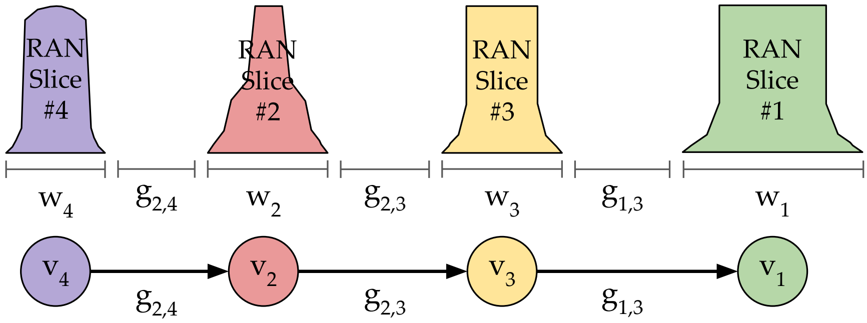


FIGURE 4.5: An example result of the FOTSP resource management optimisation problem given by solving $P3$, showing the sequence of RAN slices with minimal total isolation overhead.

of multiple disconnected paths between RAN slices (known as subtours) and ensure that the final solution is a single continuous path, i.e., a single sequence of all RAN slices and their respective required guard bands. The solution to $P3$ yields the optimal embedding of RAN slices that minimises the amount of resources consumed with guard bands, as shown in Fig. 4.5.

The advantages of modelling heterogeneous RAN slices and their requirements for isolation in the frequency domain as a complete weighted graph and embedding the RAN slices on a RANaaS platform as a FOTSP are threefold: (*i*) transparency to the type and granularity of resources from the individual RATs; (*ii*) possibility to consider the required additional guard bands between RAN slices by design; and (*iii*) extensibility to embed new types of RATs and numerologies only based on their required guard bands. However, similar to other TSPs, the inclusion of a new node (or RAN slice) requires re-running the FOTSP, an NP-complete Mixed-Integer Linear Programming problem of complexity $\mathcal{O}(n!)$, for which it is difficult to obtain the solution in reasonable time for a large number of nodes [106]. In the next subsections, we detail two heuristic algorithms for finding feasible solutions using a lower computational footprint.

4.2.4 Greedy Algorithm

The Greedy Algorithm (GA) is a simple heuristic method for solving TSPs, which can be easily implemented and serves as an initial solution to a number of improvement heuristics [108]. Nonetheless, the GA can still lead to good results when the number of nodes is small, and it is computationally cheap in comparison to more complex approaches that entail the exchange of nodes or edges to find better solutions [109]. Starting from the complete undirected weighted graph $G = (V, E)$, we sort all the edges $(i, j) \in E$ by weight in ascending order and select the one with the smallest weight.

```
Algorithm 1: Greedy Algorithm
input  : A complete weighted graph $G = (V, E)$
output : A list of edges connecting all nodes in $V$
$F \leftarrow \{\ \}$;
while $|F| < |V| - 1$ or $E \neq \{\emptyset\}$ do
    $k, l \leftarrow arg\ min(\{g_{i,j} : i, j \in V, i \neq j\})$;
    if adding $(k, l)$ to $F$ does not form a loop or branch then
        $F \overset{+}{\leftarrow} (k, l)$;
    end
    $E \setminus \{(k, l)\}$;
end
return $F$
```

If adding this edge to the solution set F will not create a loop or branch, then we include it in the solution set. Then, we remove the selected edge from E. We repeat this process of complexity $\mathcal{O}(n^2 \cdot log(n))$ until we have no more edges in E, or we have $|V| - 1$ edges in the solution set, which will contain all the nodes in V, as detailed in Algorithm 1.

4.2.5 Nearest Neighbour Algorithm Improved with 2-Opt

The Nearest Neighbour Algorithm (NNA) is a type of constructive heuristic method for solving TSPs. Starting from a random node $i \in V$, we search for the adjacent node of smallest weight. Then, we include the edge between them in F, and use the adjacent node as the next starting node. We remove the visited nodes from V and repeat this process of complexity $\mathcal{O}(n^2)$ until we have no more nodes left in V, as detailed in Algorithm 2. The NNA is computationally inexpensive, but it can often miss routes with smaller weights due to its naive and greedy nature. We can further improve the solution found with the NNA using the 2-Opt local search algorithm. It develops an initial solution and iteratively searches for improvements in the neighbourhood of that solution [105]. Starting from the initial solution from the NNA, we take two edges from the route, exchange these edges with each other, and calculate the new total weight [109]. If this exchange leads to a lower total weight, then the current route is updated. We repeat this process of additional $\mathcal{O}(n^2)$ complexity until no more improvements are found or until a predetermined number of iterations is completed [108] [110]. We refer to [109] for further information about the 2-Opt algorithm and its implementation.

Algorithm 2: Nearest Neighbour Algorithm

```
input  : A complete weighted graph $G = (V, E)$
output : A list of edges connecting all nodes in $V$
$F \leftarrow \{\ \}$;
$k \leftarrow$ random node $i \in V$;
do
    $V \setminus \{k\}$;
    $l \leftarrow arg\ min(\{g_{k,j} : j \in V, k \neq j\})$;
    $F \overset{+}{\leftarrow} (k, l)$;
    $k \leftarrow l$;
while $V \neq \{\emptyset\}$;
return $F$
```

4.3 Numerical Evaluation

In this section, we present numerical results generated by simulating the embedding of heterogeneous RAN slices using the formulation and heuristics algorithms detailed in Sec. 4.2. First, we describe the simulation setup used in our evaluation. Then, we compare the performance of the optimal solution and the two heuristic algorithms, and benchmark their computational footprint regarding optimisation time and memory consumption. For simplicity, we denote the optimal solution to $P3$ found using the Gurobi optimiser [111] by OSF.

4.3.1 Simulation Setup

We consider the scenario where an MNO wants to allocate sets ranging from 15 to 150 heterogeneous RAN slices with minimal total isolation overhead, targeting the deployment of RAN slices in a semi-persistent fashion to cover factories or sports events, and hence, only requiring the reconfiguration of the network deployment in the granularity of hours or days. Without loss of generality, we focus on RAN slices that resemble the 5G frame structure, consisting of a block of 12 OFDM subcarriers. We have run 100 iterations of the simulation, where each RAN slice can have a random combination of (*i*) numerology, a tailored subcarrier spacing for serving different types of users, e.g., stationary or on high-speed trains, in the set {15, 30, 60, 120} kHz; and (*ii*) SIR tolerance, resilience to interference according to the type of service, e.g., best effort or mission-critical, in the set {20, 25, 30, 35, 40, 45} dB [95] [96]. It is worth mentioning that the calculation of the guard band values to ensure isolation between each pair of RAN slices is outside the scope of this letter; instead, we rely on the optimal guard band values for different numerologies presented in [95].

4.3.2 Simulation Results

We consider the total isolation overhead relative to the total allocated bandwidth of RAN slices as the KPI for the embedding of heterogeneous RAN slices. In other words, this KPI represents the percentage of the total resources consumed with the guard bands between virtual radios. We compare the performance of the different embedding methods, and Fig. 4.6 shows the results of our evaluation for different numbers of RAN slices. It shows that both the GA and the NNA improved with the inclusion of 2-Opt and present performance very close to optimum, only deviating 0.98–3.85% and 0.23–0.62% from the OSF, respectively. We can also observe that the percentage of the total resources consumed with guard bands decreases in proportion to the number of RAN slices. This behaviour is due to the fact that in larger sets of different RAN slices, it is more likely that several RAN slices will possess similar characteristics in terms of RAT or numerology, which tend to require a narrower guard band between one another [94] [95].

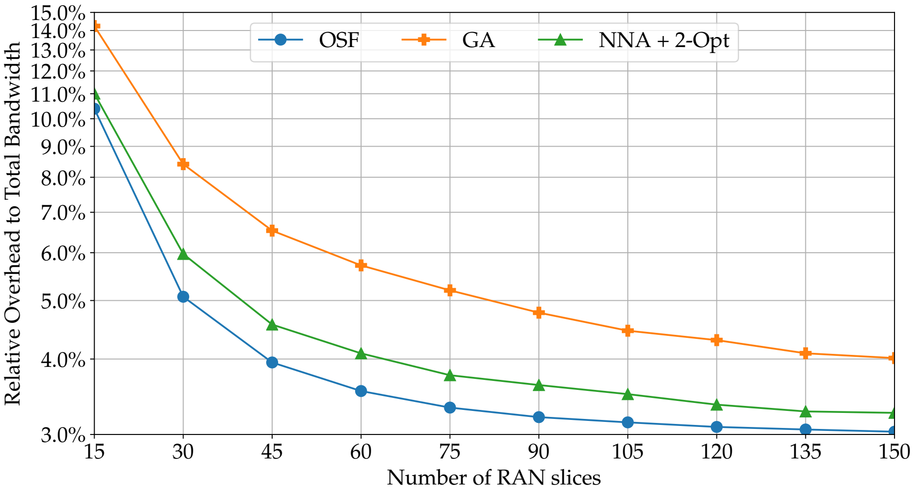
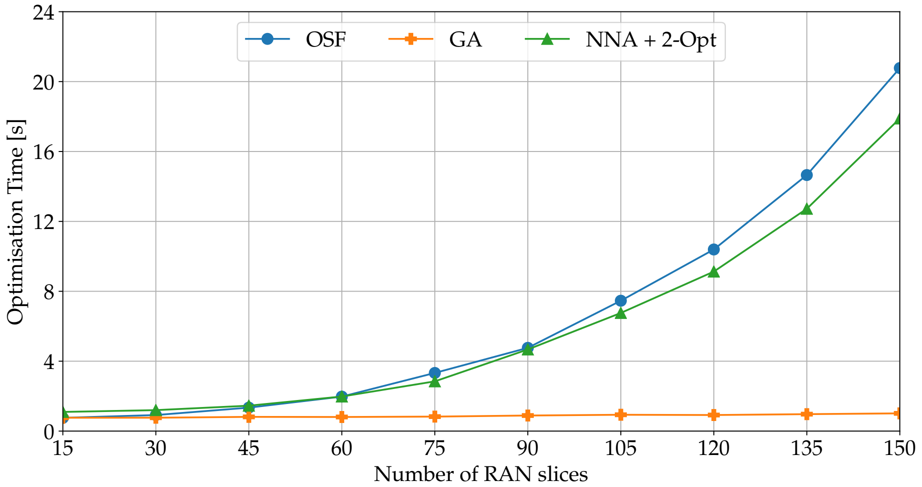
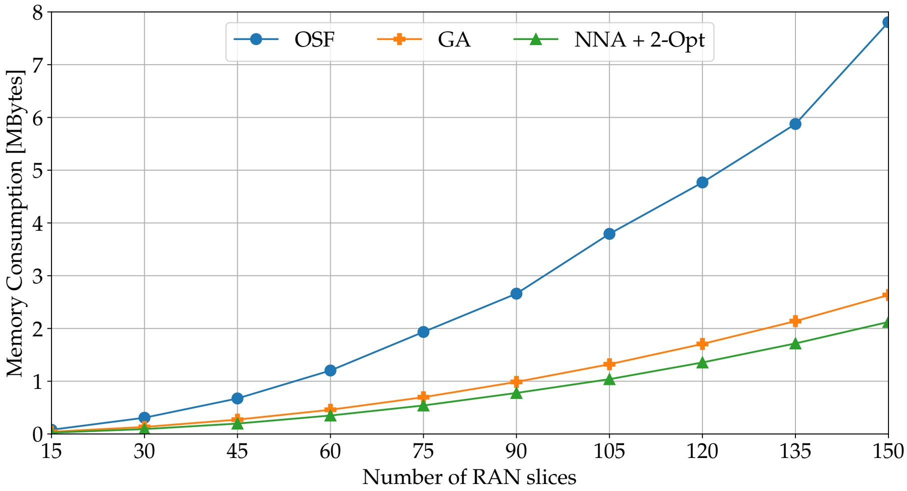


FIGURE 4.6: Performance comparison between the different embedding methods, in terms of the total isolation overhead relative to the total allocated bandwidth of RAN slices. The heuristic algorithms are considerably close to optimal and also display a similar trend according to the number of RAN slices.

We benchmarked the optimisation time and memory consumption for running the different embedding methods, and Fig. 4.7 shows the results of our measurements. Both the GA and the NNA improved with the inclusion of 2-Opt have considerably low memory usage, consuming 3 times less memory than the OSF for 150 RAN slices. The NNA improved with the 2-Opt runs faster than the OSF (16% less time for 150 RAN slices), but it also grows exponentially with the number of RAN slices, taking at least 5 seconds for embedding 90 or more RAN slices. For the same number of RAN slices, the GA can still find solutions near real-time, with its optimality gap falling to 1.56%. Therefore, we can conclude that in the timespan we consider for the deployment of RAN slices, i.e., in the order of hours and days, the optical solution should be used. Nonetheless, the NNA improved with the 2-Opt is a good solution for embedding up to 90 heterogeneous RAN slices using a lower computational footprint, while for larger numbers of RAN slices, a simpler GA could be used.



(A) The optimisation time for running the OSF and the NNA grows exponentially to the number of RAN slices, whereas the GA remains nearly constant.



(B) The memory consumption for running the different methods grows with the number of RAN slices, where both heuristics require considerably less memory than the OSF.

FIGURE 4.7: Computational footprint comparison between the different embedding methods, in terms of the optimisation time (top) and memory consumption (bottom).

4.4 Conclusions

In this chapter, we addressed the embedding of heterogeneous RAN slices for technology-agnostic RANaaS. First, we introduced a novel formulation using a graph representation of RAN slices in RANaaS platforms, which is transparent to the resources of the different RAN slices, considers the required additional guard band by design, and is extensible to support new RATs or numerologies. Then, we proposed a FOTSP-based resource management optimisation problem that leverages the characteristics of the virtual radios and their required isolation for determining the optimal arrangement of RAN slices that minimises the total isolation overhead, saving radio resources and increasing the overall efficiency of the RANaaS platform. We compared different methods for solving the FOTSP, and our simulation results suggest that the optimal solution should be used in our scenario of interest, with the deployment and reconfiguration of RAN slices in granularity of hours and days. However, a combination of heuristics algorithms provide a good solution for the problem using a lower computational footprint. The NNA improved with the 2-Opt is suitable for embedding a smaller number of RAN slices, with the GA becoming more attractive as the quantity of RAN slices increases.

Chapter 5

Orchestrating Multiple Network Segments to Create E2E NSs

Orchestrating Multiple Network Segments to Create E2E NSs

> *We are only as strong as we are united,*
> *as weak as we are divided.*
>
> Albus Dumbledore, Harry Potter and the Goblet of Fire, *2000*

IN this chapter, we present the vision of the Orchestration and Reconfiguration Control Architecture (ORCA) Horizon 2020 project, and propose a hierarchical orchestration architecture for E2E networks. Our proposal addresses the oversimplified resource allocation and limited support for network segments of existing E2E orchestration solutions, by leveraging domain expertise and enabling the independent management of each segment, using a combination of distributed specialised orchestrators already deployed to manage the network infrastructure, as shown in Fig. 5.1. We introduce a higher-level orchestrator, the hyperstrator, to coordinate the distributed orchestrators and deploy NSs across multiple network segments. To the best of our knowledge, this is the first solution that decentralises both the control over the physical infrastructure and the decision over the resource management for E2E networks. These approaches facilitate upgrading or replacing the existing underlying orchestrators, and including new segments or types of resources unforeseen at design time. Without loss of generality, in this chapter we focus on NSs comprised of RAN and CN segments, interconnected by a TN segment, in the administrative domain of an MNO.

The technical work presented in this chapter is based on our works “Orchestrating Next-Generation Services Through End-to-End Network Slicing” released as a white paper from the Horizon 2020 ORCA project [112], and “Breaking Down Network Slicing: Hierarchical Orchestration of End-to-End Networks” published in IEEE Communications Magazine [113].

The remainder of this chapter is organised as follows. In Section 5.1, we introduce our hierarchical orchestration architecture for E2E networks, leveraging the distributed intelligence of specialised orchestrators to achieve a fine-grained resource allocation across multiple network networks, while ensuring coordination through a higher-level orchestrator, the hyperstrator. In Section 5.2, we detail the challenges and design choices for realising a prototype of our hierarchical orchestration architecture. In addition, we validate its ability to deploy customised NSs by showing the NS deployment and the impact of the resource allocation per network segment on the performance of E2E NSs. Finally, in Section 5.3, we conclude this chapter and pose our final remarks.

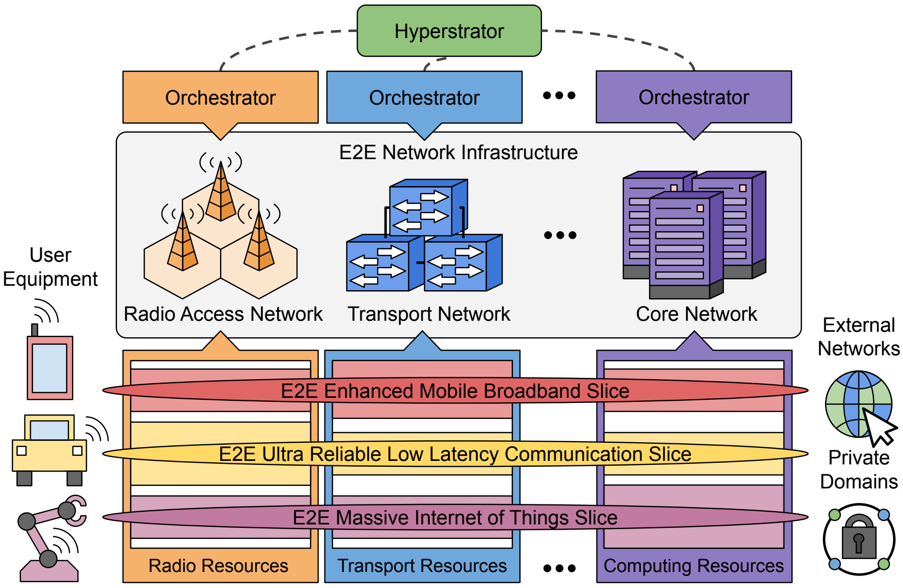


FIGURE 5.1: Our proposed E2E network design with one specialised orchestrator per network segment, and a hyperstrator for coordinating the resource allocation across segments.

5.1 Hierarchical Orchestration of E2E Networks

In this section, we propose a hierarchical orchestration architecture for E2E networks, using a set of distributed specialised orchestrators for managing different network segments, as shown in Fig. 5.2. Each existing orchestrator is responsible for the resource management in a particular network segment, while we coordinate the resource allocation and function placement across orchestrators and their respective network segments through a higher-level orchestrator, namely, a hyperstrator. Our hierarchical orchestration architecture enables the decentralisation of the control and decision over E2E networks, breaking down the E2E resource management and network slicing problems into smaller, tractable problems per network segment.

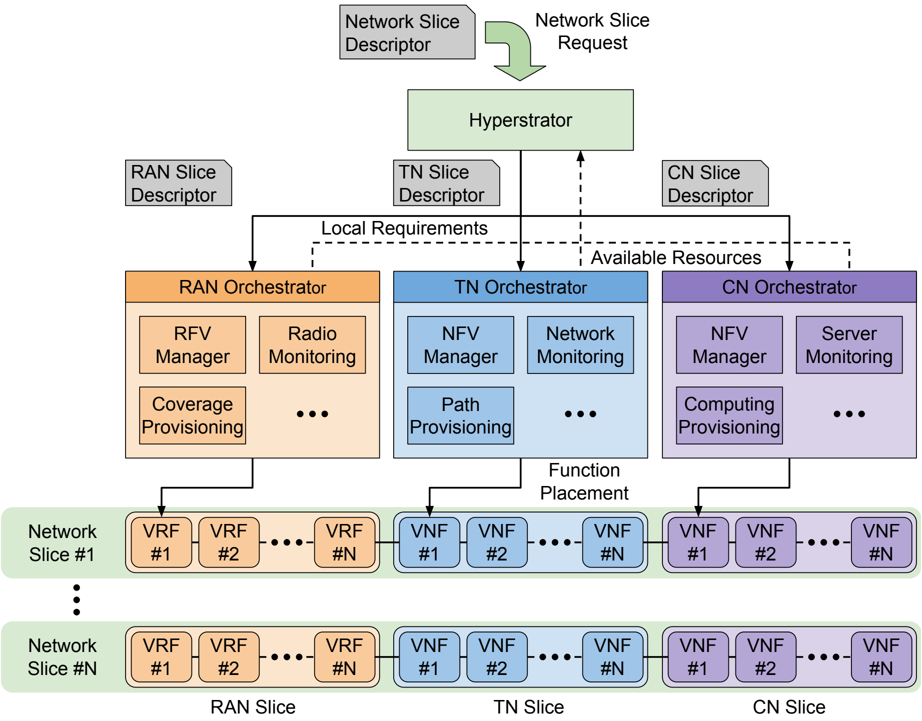


FIGURE 5.2: Our hierarchical orchestration architecture, where each network segment already has its own specialised orchestrator, and we accomplish cross-network segment orchestration through a new entity, the hyperstrator.

In the remainder of this section, we introduce the hyperstrator and detail how it leverages multiple specialised orchestrators for managing E2E networks, while ensuring cohesive E2E resource allocation across network segments.

5.1.1 Coordinating Distributed Network Orchestrators

The creation of NSs imposes both local requirements for specific network segments, e.g., coverage areas on RAN segments, or points of exchange on TN segments; and global requirements for the entire E2E network, e.g., throughput, delay, and reliability targets that the combination of network segments must meet to comply with the QoS of the NSs. Due to the different purposes and independence between network segments, we can separate the E2E resource management per network segment as long as we ensure that each network segment delivers the necessary performance for meeting the global requirements [25] [78]. We leverage this separation to decentralise the orchestration of E2E networks, delegating the decision over the resource allocation and function placement for each network segment to a respective specialised orchestrator. Such compartmentalisation allows each individual specialised orchestrator to focus on a limited number of well-defined tasks, using models and paradigms tailored for the particularities of its respective network segment. This approach facilitates achieving a fine-grained resource allocation in every network segment, while reducing the resource management complexity, both in terms of design and implementation.

This decentralised orchestration paradigm requires coordination between different orchestrators for deploying NSs, a process that involves dimensioning, creating, and combining multiple NSSs. To do so, the distributed specialised orchestrators could interact amongst themselves through east-westbound interfaces following a Self Organising Network (SON) model. However, this approach would require each orchestrator to (*i*) implement new communication interfaces; (*ii*) be aware of the existence and capabilities of other orchestrators; and (*iii*) know how to negotiate the use of their resources. This would require extensive modifications and further development on the existing orchestrators, while also introducing considerable communication and negotiation overheads. Rather than taking this costly approach, we propose the introduction of a new entity above the distributed specialised orchestrators, the hyperstrator, which is in charge of interacting with the underlying orchestrators and coordinating the lifecycle of NSs. In this way, not only each specialised orchestrator becomes oblivious to the existence of other orchestrators and network segments, but this approach also allows us to potentially integrate existing orchestrators present in real network deployments under the hyperstrator. We can achieve this by developing simple, bespoke translation layers between existing northbound configuration interfaces (used by MNOs to configure their networks) and the hyperstrator.

The combination of the hyperstrator and the distributed specialised orchestrators acts as a single E2E network orchestrator, responsible for managing the entire E2E network infrastructure in the administrative domain of an MNO. The hyperstrator itself is the interface for interacting with the E2E network, the central point for (*i*) instantiating customised NSs leveraging heterogeneous resources available across multiple network segments; (*ii*) monitoring the existing communication services by gathering KPIs from the distributed orchestrators; and if necessary, (*iii*) optimising the use of network resources according to the demands of communication services. Therefore, the hyperstrator serves as a network automation tool that facilitates the MNO's network operations, simplifying the deployment of communication services and supporting the offer of NSaaS, by automatically provisioning NSs to serve tenants [10]. In addition, the hyperstrator in charge of one administrative domain could act as a tenant to lease resources from the hyperstrators responsible for other administrative domains, e.g., requesting RAN slices to provide coverage in different geographical locations, or CN slices to offload computations during peak network demand. In this regard, using a hyperstrator per administrative domain could facilitate and standardise resource sharing among MNOs.

5.1.2 Translating and Delegating E2E Requirements

The tenants can request NSs to the hyperstrator, specifying them using NS descriptors, manifests containing high-level E2E requirements [16], e.g., mobile coverage on particular geographical locations with specific connectivity to certain data centres. The NS descriptors can potentially be implemented using JSON and YAML formats, adopting YANG or TOSCA modelling languages [32]. In parallel to NFVO-C [78], the hyperstrator is responsible for translating these high-level E2E requirements into requirements for specific network segments, in the form of NSS descriptors, e.g., RAN and CN slice descriptors. However, in contrast to NFVO-C, the NSS descriptors are tailored to the capabilities of each network segment [32], e.g., containing the required throughput, latency and reliability, as well as the coverage areas and points of exchange, respectively. The MNOs can specify or develop different translation mechanisms for their hyperstrators, such as the one detailed in [16]. After the translation procedure is finished, the hyperstrator forwards the NSS descriptors to the respective underlying specialised orchestrators, as shown in Fig. 5.2. Each orchestrator is free to use its own resource management directives to map local requirements into low-level network configuration for creating suitable NSSs, which involves deciding the appropriate allocation of resources and placement of functions to fulfil the local requirements [114].

The resulting E2E NSs may comprise multiple NSSs, each of which can possess chains of different types of functions, e.g., VNFs, implementing network services and upper layers of the communication stack; and Virtual Radio Functions (VRFs), implementing RATs and lower layers of the communication stack [34] [79]. Depending on the resources allocated and the functions placed, each NSS will achieve different performance. The hyperstrator can guarantee consistent QoS for the E2E NSs by requesting NSSs that deliver the required E2E throughput, while remaining within the E2E delay and reliability budgets of the E2E NS. However, some NSSs may have resource and performance limitations, which the hyperstrator can attempt to circumvent by realising a trade-off in the E2E resource allocation and requesting NSSs with more resources, and that deliver higher performance, from other network segments.

5.1.3 Ensuring Consistency Across Multiple Network Segments

The distributed nature of our hierarchical orchestration architecture poses new challenges for designing and implementing this solution, especially for the hyperstrator, which must ensure consistency between multiple specialised orchestrators and network segments. The system must cope with issues that may arise during the instantiation of E2E NSs across different segments, e.g., lack of the necessary resources, failures in the resource allocation or functional placement, as well as unexpected communication and hardware failures. We can circumvent such failures using a transactional communication protocol between the hyperstrator and the underlying specialised orchestrators, where: the hyperstrator requests the execution of operations on multiple network segments, and according to the results of such requests, the hyperstrator either commits the operations, making the orchestrators carry out the commands, or rolls them back, reverting the instantiation of NSSs. This approach enables persistent, atomic operations over the management of E2E NSs, while ensuring consistency across multiple network segments [115]. Furthermore, the E2E NS should possess a unique identifier across all networks segments, e.g., Universally Unique Identifier (UUID), facilitating any operation over the NSs and access to their information, as well as the implementation of authorisation mechanisms for managing the NSs, such as Access Control Lists (ACLs).

Our proposed architecture differs from existing hierarchical orchestration approaches in the literature, as we do not centralise the intelligence and decision over the E2E network management in a single monolithic entity, responsible for orchestrating certain types of network segments [30] [31] [75] [78]. In reality, the hyperstrator does not have any role in the resource allocation and function placement on particular network segments. Instead, the hyperstrator (*i*) translates global service requirements for creating NSs into local requirements for specific network segments; (*ii*) delegates local requirements to the respective underlying specialised orchestrators, which are free to adopt the state-of-the-art or proprietary resource management solutions for creating customised NSSs; and most importantly, (*iii*) ensures cohesive performance across NSSs to guarantee a consistent QoS for the NSs. In addition, it is worth mentioning that our solution targets existing network deployments, focusing on the integration with the orchestrators already deployed to manage the MNO's network infrastructure, unlike clean slate solutions like CORD and ONAP that require a complete replacement of the existing control and orchestration entities. Therefore, our hierarchical orchestration architecture leverages the capabilities of existing specialised orchestrators and the domain expertise of their established communities for providing the most effective resource management in every network segment.

5.2 Experimental Proof of Concept

In this section, we detail an example deployment of E2E NSs using our hierarchical orchestration architecture, with the hyperstrator coordinating separate specialised orchestrators responsible for different network segments. First, we describe a proof-of-concept implementation of our hierarchical orchestration architecture used for managing an experimental E2E network infrastructure. Then, we assess the overhead introduced by the distributed nature of our architecture to provision E2E NSs, as well as the impact of each NSS on the performance of E2E NSs.

5.2.1 Building the Experimental Setup

To verify the feasibility of our proposed architecture, we have created an experimental E2E network infrastructure that resembles mobile networks, consisting of RAN, TN and CN segments managed by a hierarchy of orchestrators, as illustrated in Fig. 5.3. The conjunction of these network segments connects clients with their UEs to desired microservices in the CN. The prototype hyperstrator serves as the central point of management for our E2E network infrastructure and the deployment of E2E NSs. It provides a Create, Read, Update, Delete (CRUD)-based interface, where tenants can instantiate, query, modify or remove NSs. The tenants must specify the requirements of their NSs in the form of NS slice descriptors, JSON data structures that contain high-level E2E requirements. In possession of such information, our experimental proof of concept operates as follows.

- The hyperstrator translates the high-level E2E requirements into requirements specific for each network segment. In this case, considerations include coverage to UEs in the RAN, computing to host services in the CN, and paths between both network segments in the TN.
- Then, the hyperstrator requests the instantiation of NSSs with tailored requirements to the underlying orchestrators. Each orchestrator is free to adopt their own resource management directives, protocols and internal processes to allocate resources and place functions.

- Upon successful creation of the necessary NSSs, the NS becomes operational and the UEs can start communicating with a new microservice. The hyperstrator returns the UUID of the NS to the tenant, who can use this identifier to operate over the NS.

In Appendix B, we detail the hyperstrator's interfaces for managing E2E NSs, including both its northbound interface towards tenants and its southbound interface towards the distributed specialised orchestrators.

To simplify the development of our initial prototype, we employed the ZMQ messaging library [85] to implement both the northbound communication with tenants and the southbound interface towards the underlying orchestrators. In addition, we created homebrewed orchestrators for managing each network segment, responsible for the basic dimensioning of required resources to meet local service requirements, as well as the communication with the hyperstrator and the respective underlying controllers. In the future, we plan to extend ONOS and OSM, adding new capabilities for supporting our hierarchical orchestration architecture, and use them to replace our homebrewed orchestrators. In the development of our experimental E2E network infrastructure, we used Linux Containers (LXC) [116] on servers at the Iris Testbed (`http://iristestbed.eu/`) for realising the majority of the elements in our setup, reducing its physical size and overall complexity. Furthermore, we targeted using a RAT compatible with a broader range of UEs, which motivated us to choose WiFi for our evaluation. However, WiFi has stringent timing requirements for its MAC scheme (in the order of nanoseconds), which prevented us for using the combination of XVL and HyDRA detailed in Chapter 3 (which introduces a service delay in the order of milliseconds, as demonstrated in Section 3.4.4). Instead, we relied on OpenWiFi [117], a controller for SDRs that implements a technology-specific radio virtualisation mechanism for slicing WiFi RATs.

We leveraged software-defined hardware platforms as enablers for network slicing [75], and have set up an SDR-based RAN segment, an SDN-based TN segment, and an LXC-based CN segment, as shown in Fig. 5.3. The RAN segment is composed of a Zynq SDR running the OpenWiFi SDR controller, allowing us to create a Hostapd access point with tailored resource allocation for each UE. We create RAN slices using OpenWiFi's frame-based radio virtualisation mechanism to allocate non-overlapping portions of airtime for meeting local requirements. The TN segment is composed of four Open vSwitch [118] containers forming an emulated ring topology in a virtualisation server. Two of these switches are also attached to Ethernet ports on the Zynq SDR and the cloud server,

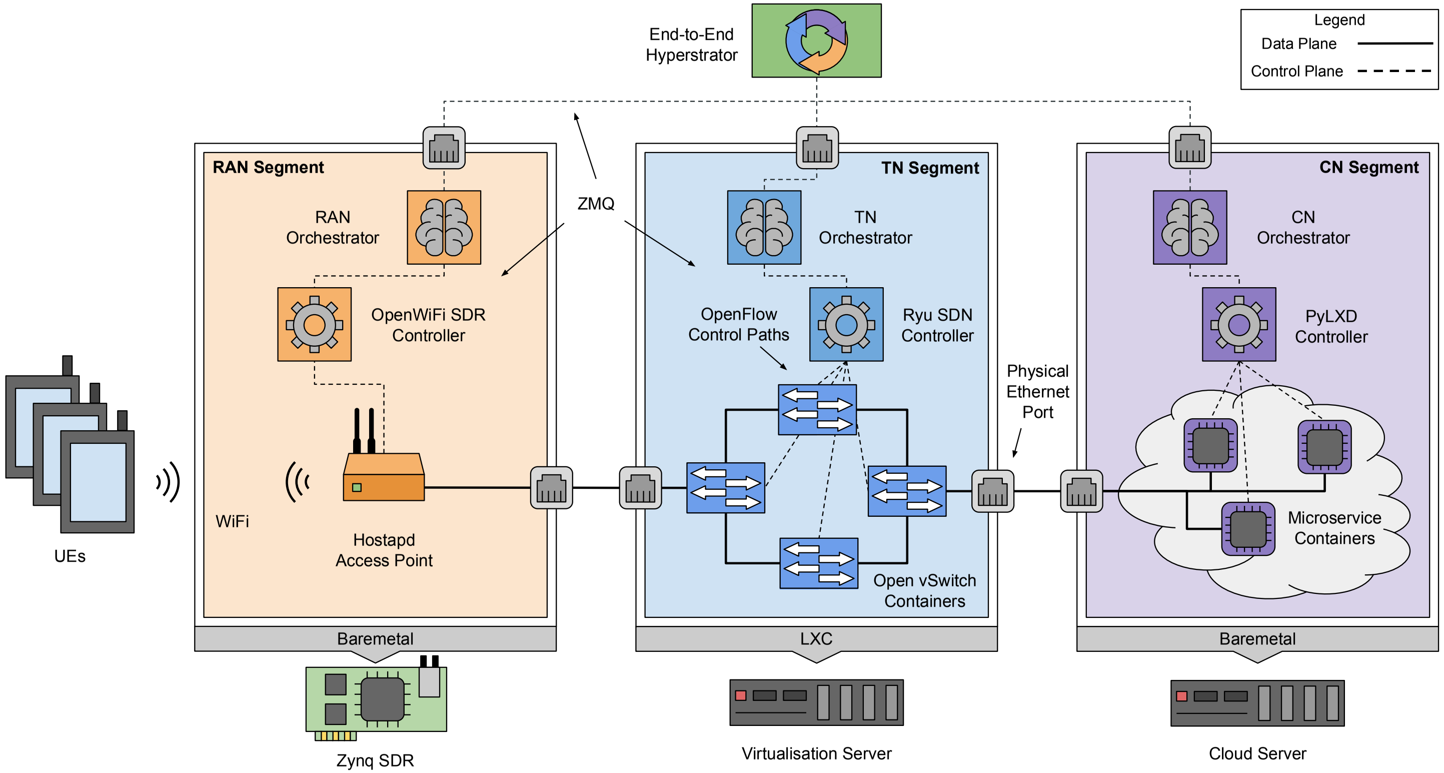


FIGURE 5.3: The experimental setup we developed to validate our hierarchical orchestration architecture. Each network segment possesses its own specialised orchestrator, while we coordinate the resource allocation across network segments for creating E2E NSs through a proof-of-concept implementation of the hyperstrator.

serving as points of exchange towards the RAN and CN segments, respectively. We create TN slices using Ryu SDN controller's features for creating overlay networks with isolated traffic and tailored queue configurations [119]. The CN segment is composed of a cloud server running LXD [116], allowing us to host microservices on containers with customised computing specifications. We create CN slices using PyLXD controller's features for instantiating customised containers set up with different Operating System (OS) images and tailored computing resources [120].

To combine our NSSs, we attribute IP addresses to new RAN and CN slices, and then use new TN slices to interconnect them. In the RAN segment, we programmatically configure a DHCP server running in the Zynq SDR for assigning valid IP addresses to new UEs and forward their traffic to the data plane Ethernet port. In the CN segment, for each container hosting a new service, we create a virtual network interface attached to a bridge network with automatic DHCP, which is connected to the cloud server's data plane Ethernet port. Then, we use the IP addresses of the RAN and CN slices as arguments for creating TN slices, where we calculate and establish the shortest path between all the routes that support the required throughput between these IPs. In addition, we utilise *iptables* for blocking communication between UEs and between containers, ensuring traffic isolation within the same segment. For further information, the reader can refer to the hyperstrator's public repository (`https://github.com/orca-project/hoen`).

5.2.2 Overhead for Provisioning Network Slices

In this analysis, we are interested in the interaction between the elements of our hierarchical orchestration architecture and their delay for provisioning E2E NSs across multiple network segments. The total provisioning delay consists of the time interval required for deploying new E2E NSs to support communication service, i.e., between tenants sending NS requests until the NSs are fully available, which includes the translation of high-level E2E requirements, the dimensioning of NSSs, and the allocation of resources on all network segments. Fig. 5.4 shows the results of our measurements for deploying customised NSs using our hierarchical orchestration architecture. The hyperstrator took 2.2 seconds to fulfil the NS request and instantiate a new E2E NS. After receiving the NS request, the instantiation of RAN, TN and CN slices accounted for 2.99%, 0.84% and 95.85% of the provisioning delay, respectively. The hyperstrator functionality and the sequential communication with the distributed orchestrators accounted for roughly 2 ms, adding an overhead of 0.08%. We could further reduce the total provisioning delay by allocating NSSs concurrently or pre-allocating containers, which would reduce it to 90 ms and allow the instantiation of up to 11 NSs per second. These results show that the hyperstrator can orchestrate heterogeneous network resources for provisioning E2E NSs near real-time, and that the distributed nature of our proposed architecture introduces a negligible overhead regarding the total provisioning delay.

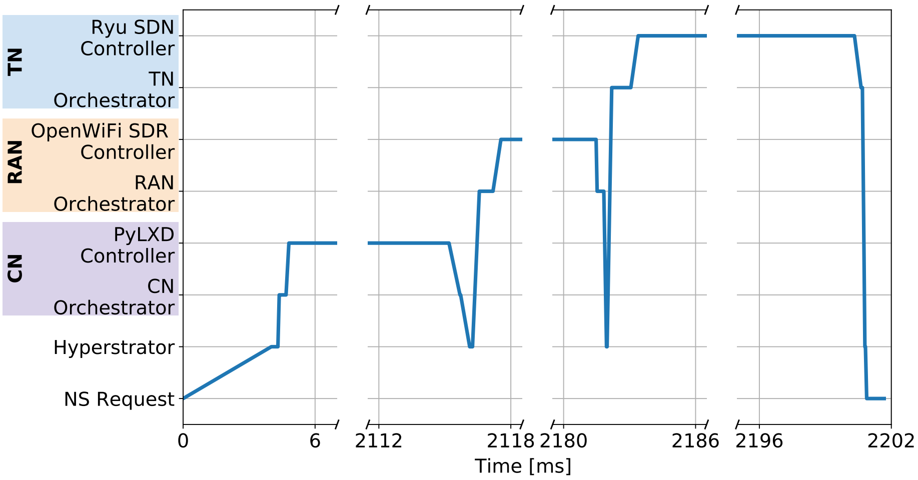


FIGURE 5.4: Timing diagrams showing the interactions between the entities of our experimental setup required for deploying NSs.

5.2.3 Consistent QoS across Network Segments

In this analysis, we assess how the resource allocation in each network segment affects the E2E performance of the NSs. Due to the independence between different network segments and their resource management directives, the hyperstrator must coordinate the underlying specialised orchestrators for deploying NSs with consistent QoS across multiple network segments. Table 5.1 shows the E2E performance in terms of throughput and round-trip delay for an E2E NS traversing through the CN, TN, and RAN segments towards a UE. For each network segment, we varied the amount of allocated resources, i.e., radio airtime in the RAN segment, link capacity in the TN segment, and CPU cycles in the CN segment, while allocating the maximum resources in the other network segments; and measured the experienced E2E performance between a UE and its microservice container. These results illustrate how the resource allocation in each NSS significantly impacts the E2E performance of NSs, motivating the need for coordination between the distributed specialised orchestrators in charge of the different network segments to ensure consistent QoS for the E2E NSs.

	RAN		TN		CN	
	Radio Airtime		Link Capacity		CPU Cycles	
	Throughput [Mbps]	Delay [ms]	Throughput [Mbps]	Delay [ms]	Throughput [Mbps]	Delay [ms]
100%	21.8	1.97	22.3	2.41	24.5	16.982
90%	17.9	2.10	22.2	1.86	**23.2**	**41.67**
80%	16.8	3.09	22.0	2.10	20.5	105.10
70%	14.1	4.35	22.7	1.75	18.6	147.59
60%	**11.5**	**6.48**	22.7	2.07	16.1	191.16
50%	8.47	8.07	21.5	2.18	13.2	219.32
40%	6.41	19.41	21.3	2.13	11.2	372.04
30%	5.44	35.24	22.2	2.14	8.40	466.51
20%	4.02	43.40	19.8	2.04	6.13	639.77
10%	2.18	61.37	18.7	2.01	3.03	875.99
1%	0.43	124.27	**9.88**	**1.98**	1.20	998.40

TABLE 5.1: Example of how the allocation of distinct types of resources in different network segments affects the E2E performance of the NSs. We can use the hyperstrator to coordinate the different orchestrators to establish a dedicated 10 Mbps E2E NS with latency of 50 ms using 60% of the radio resources, 1% of the transport resources, and 90% of the computing resources (shown in bold), while still leaving available resources for creating more E2Es NSs tailored to serve other services.

5.3 Conclusions

In this chapter, we addressed the oversimplified resource allocation and limited support for network segments of existing E2E orchestration solutions. We presented a hierarchical orchestration architecture for E2E networks, breaking down the E2E network management and network slicing challenges per network segment, while leveraging the capabilities and the established communities of existing orchestrators to achieve an E2E fine-grained resource allocation. It is a paradigm shift from traditional E2E network orchestration solutions like ONAP, which operate in a clean slate manner and centralise the intelligence for managing entire E2E network infrastructures in single monolithic entity. Our proposal: (*i*) leverages domain expertise and enables the independent management of each network segment using the existing distributed specialised orchestrators already deployed to manage the network infrastructure; (*ii*) coordinates the resource management across network segments through a higher-level orchestrator, the hyperstrator; and (*iii*) is both modular and extensible, capable of supporting new types of network segments and resources unforeseen at design time. We developed a proof-of-concept implementation of our hierarchical orchestration architecture and evaluated its delay for provisioning NSs, as well as the impact of the resource allocation per network segment on the performance of NSs. Our results show that the distributed nature of our solution introduces negligible overhead to provision E2E NSs in our experimental E2E network infrastructure, confirmed the need for a hyperstrator to coordinate different network segments and ensure consistent QoS for E2E NSs, and validated that we can streamline the coordination of multiple specialised orchestrators already deployed in real networks to instantiate E2E NSs.

Chapter 6

Conclusions and Future Directions

Conclusions and Future Directions

> “Don’t adventures ever have an end? I suppose not.
> Someone else always has to carry on the story.”
>
> Bilbo Baggins, The Fellowship of The Ring, *1954*

THE main objective of this PhD thesis was to explore the potential of radio virtualisation in the context of E2E network slicing. First, to leverage existing radio virtualisation solutions to provide RANaaS. Then, to model the resource allocation of virtual radios from heterogeneous RAN slices on RANaaS platforms. Finally, to ensure a consistent QoS for E2E NSs traversing through multiple networks segments. In the remainder of this chapter, we summarise the key research contributions of this PhD thesis, and propose research directions inspired by the topics we addressed. The high-level outline of this thesis is depicted in Fig. 6.1.

6.1 Summary of Contributions

In Chapter 3, we first investigated the functionality and challenges for enabling RANaaS through radio virtualisation, and identified the key requirements for using radio hypervisors to support RANaaS. We provided an answer to the RQ1: *"What are the requirements for using radio hypervisors to support RANaaS?"*, by compiling a comprehensive list of features that radio hypervisors must incorporate for supporting RANaaS. However, we observed that the majority of radio hypervisors in the literature lack some of these essential features. Then, we addressed the RQ2: *"How can radio hypervisors provision and instantiate custom virtual radios on demand?"*, by developing XVL, a modular software layer that can wrap around existing radio hypervisors and provide them with the missing resource management functionality for enabling RANaaS. We integrated XVL with a

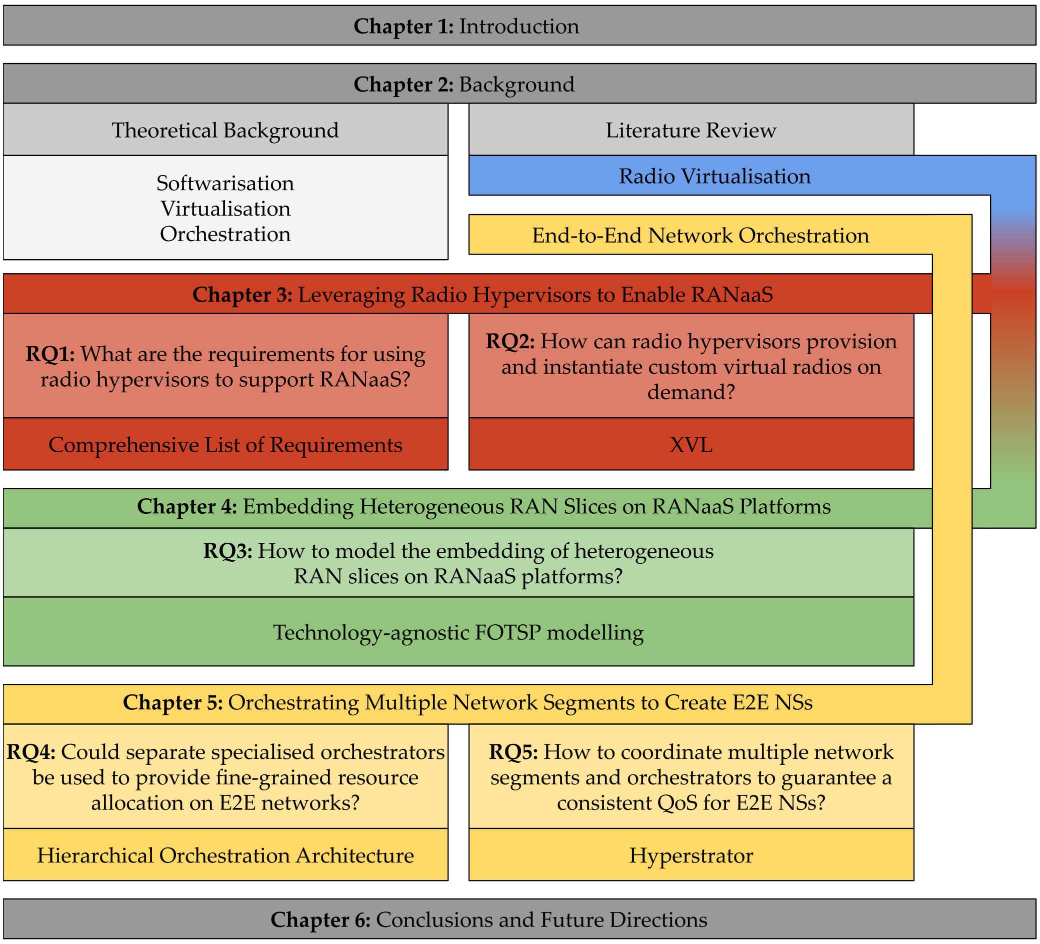

Chapter 1: Introduction
Chapter 2: Background
Theoretical Background
Softwarisation
Virtualisation
Orchestration
Literature Review
Radio Virtualisation
End-to-End Network Orchestration
Chapter 3: Leveraging Radio Hypervisors to Enable RANaaS
RQ1: What are the requirements for using radio hypervisors to support RANaaS?
RQ2: How can radio hypervisors provision and instantiate custom virtual radios on demand?
Comprehensive List of Requirements
XVL
Chapter 4: Embedding Heterogeneous RAN Slices on RANaaS Platforms
RQ3: How to model the embedding of heterogeneous RAN slices on RANaaS platforms?
Technology-agnostic FOTSP modelling
Chapter 5: Orchestrating Multiple Network Segments to Create E2E NSs
RQ4: Could separate specialised orchestrators be used to provide fine-grained resource allocation on E2E networks?
RQ5: How to coordinate multiple network segments and orchestrators to guarantee a consistent QoS for E2E NSs?
Hierarchical Orchestration Architecture
Hyperstrator
Chapter 6: Conclusions and Future Directions

FIGURE 6.1: The outline of this PhD thesis.

technology-agnostic radio hypervisor and validated its ability to deploy virtual radios with different RATs or numerologies on-demand. In addition, to the best of our knowledge, we were also the first to quantify the delay and signal degradation introduced by radio virtualisation.

In Chapter 4, we identified that current models and solutions for embedding RAN slices are not suitable for embedding virtual radios with different RATs or numerologies. We provided an answer to the RQ3: *"How to model the embedding of heterogeneous RAN slices on RANaaS platforms?"*, by introducing a graph representation of RAN slices that is technology agnostic and considers the required additional guard band between different virtual radios by design. To the best of our knowledge, we were also the first to propose and formulate a FOTSP-based resource management optimisation problem for allocating heterogeneous RAN slices on RANaaS platforms. Our novel formulation leverages the characteristics of different virtual radios and their required isolation for determining the optimal arrangement of RAN slices that minimises the total isolation overhead, saving radio resources and increasing the overall efficiency of the RANaaS platform.

In Chapter 5, we addressed the oversimplified resource allocation and limited support for network segments of existing one-size-fits-all orchestrators for E2E networks. We answered the RQ4: *"Could separate specialised orchestrators be used to provide fine-grained resource allocation on E2E networks?"*, by proposing a hierarchical orchestration architecture that breaks down the E2E network management and allows the use of existing separate orchestrators per network segment. However, this decentralised orchestration paradigm requires coordination between different orchestrators for deploying E2E NSs across multiple network segments. Then, we addressed the RQ5: *"How to coordinate multiple network segments and orchestrators to guarantee a consistent QoS for E2E NSs?"*, by proposing a higher-level orchestrator, the hyperstrator, to coordinate the composition of E2E NSs and ensure a cohesive resource allocation across multiple network segments. To the best of our knowledge, we are the first to decentralise both the control over the network infrastructure and the decision over the resource management for E2E networks. Finally, we also confirmed that our hierarchical orchestration architecture introduces a negligible overhead for instantiating E2E NSs.

6.2 Future Directions

In this PhD thesis, we made several contributions regarding the utilisation of radio virtualisation to enable E2E network slicing. In the remainder of this section, we present a number of open challenges and potential future research directions that stem from the topics we addressed.

6.2.1 Granularity of Virtual Radio Resources

Every radio virtualisation mechanism possesses a certain radio resource granularity, i.e., the minimal step or quantity of resources that the radio hypervisor can allocate. Virtual radios that require any given amount of radio resources are provided with an integer multiple of this minimal step. Limited by physical radio hardware constraints, smaller steps allow more fine-grained control over resource allocation at the cost of an increased computational cost. The resource allocation granularity in a RANaaS platform is a crucial parameter, as a mismatch between the amounts of requested and allocated resources leads to signal distortions, which can degrade or compromise the operation of the RAN slices [21] [74]. For FFT-based radio hypervisors where the IFFT size dictates the resource allocation granularity in frequency (Δf), a larger IFFT reduces the FFT bin width for a fixed B, consequently reducing the signal distortions experienced by the virtual radios. However, the degradations cannot be eliminated unless there is a perfect match between the requested and allocated bandwidths, as demonstrated in Table 3.2.

A RANaaS platform could potentially adjust the IFFT size of the radio hypervisor and the B of the physical radio hardware to find a given Δf that perfectly matches the required bandwidth of a specific virtual radio. However, it may be infeasible to find a combination of parameters that simultaneously satisfies multiple virtual radios with different configurations. Alternatively, this mismatch in the resource allocation can be mitigated using a fractional resampler per virtual radio, coupling the different sampling rates associated with the requested and allocated bandwidths at the cost of extra computations [74]. A potential future work can investigate the trade-offs for the granularity of virtual radio resources, assessing the ideal values for the IFFT size and B to accommodate different RATs and numerologies, as well as evaluate the benefits and drawbacks for using fractional resamplers. The outcome of this work would be the optimal strategy to support virtual radios with minimal signal distortions, using tailored parameters, the adoption of fractional resamplers, or a combination of both depending on the requirements of the set of virtual radios.

6.2.2 High-level RAT Requirement Translation

As part of the offer of RANaaS, the MNOs must decide how to efficiently allocate communication services onto RAN slices, which may be impractical on a one-to-one basis due to the wide range of different service requirements and low-level network configurations for the virtual radios [94]. A possible approach for MNOs to circumvent this limitation would be to group the different communication services into sets of similar performance requirements, and then find an appropriate low-level network configuration that supports each set of communication services with acceptable performance [94]. However, the challenge of mapping high-level RAT requirements, e.g., throughput, moving speed, and the number of users, into the low-level network configuration, e.g., subcarrier spacing, cyclic prefix duration, frame types [114], for the realisation of RAN slices persists. A potential future work can examine the problem of selecting a particular virtual radio configuration for supporting a given group of communication services, assessing the relationship between the degrees of freedom of flexible 5G numerologies and their effect on the KPIs of the communication services. The outcome of this work would be a mathematical framework that provides the acceptable ranges of values for a low-level network configuration to be able to accommodate a given communication service, providing insights into the effective parameters and necessary adjustments for using customised RAN slices radios to support groups of communication services with similar requirements.

6.2.3 Distributed Modelling of E2E Networks

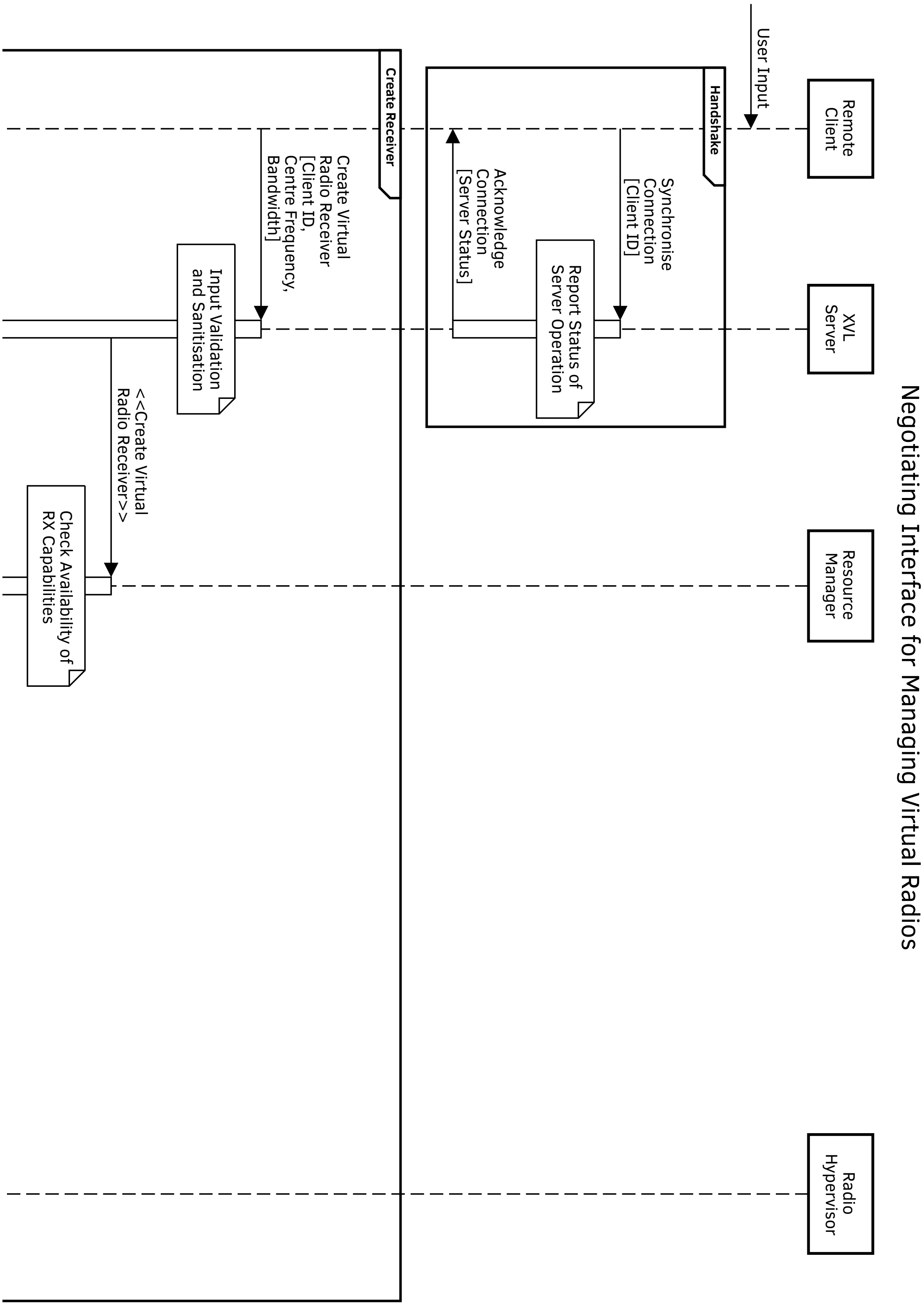
Large-scale mobile network infrastructures possess thousands of different network equipment, ranging from base stations to optical switches and computing servers. In order to efficiently use such assets to serve hundreds of thousands of users, MNOs optimise their resource allocation to minimise deployment and operational costs [121]. However, accurately modelling and optimising the resource allocation in a single network segment is already challenging and time-consuming [122] [123], an issue that is only aggravated in E2E networks comprising many different network segments, each with their own characteristics, purposes, and paradigms. Hence, a global model and optimisation of the resource allocation in an entire E2E network, including the interplay between different network segments, may be intractable or infeasible. A potential future work can investigate the distributed modelling of the resource allocation for E2E networks, breaking down the global resource allocation optimisation problem into smaller and tractable ones on a per-network segment basis. It could model E2E network infrastructure as a system of systems [124], combining the resources and functionality of the different networks segments to deploy E2E NSs, using the mathematical framework of heterogeneous network graphs to represent each network segment and the relationship between them as separate graphs [125]. The outcome of this work would show how variations in the characteristics and topology of each network segment would directly affect the performance of the global resource allocation optimisation problem; as well as allow the comparison between global and distributed resource allocation strategies for E2E networks.

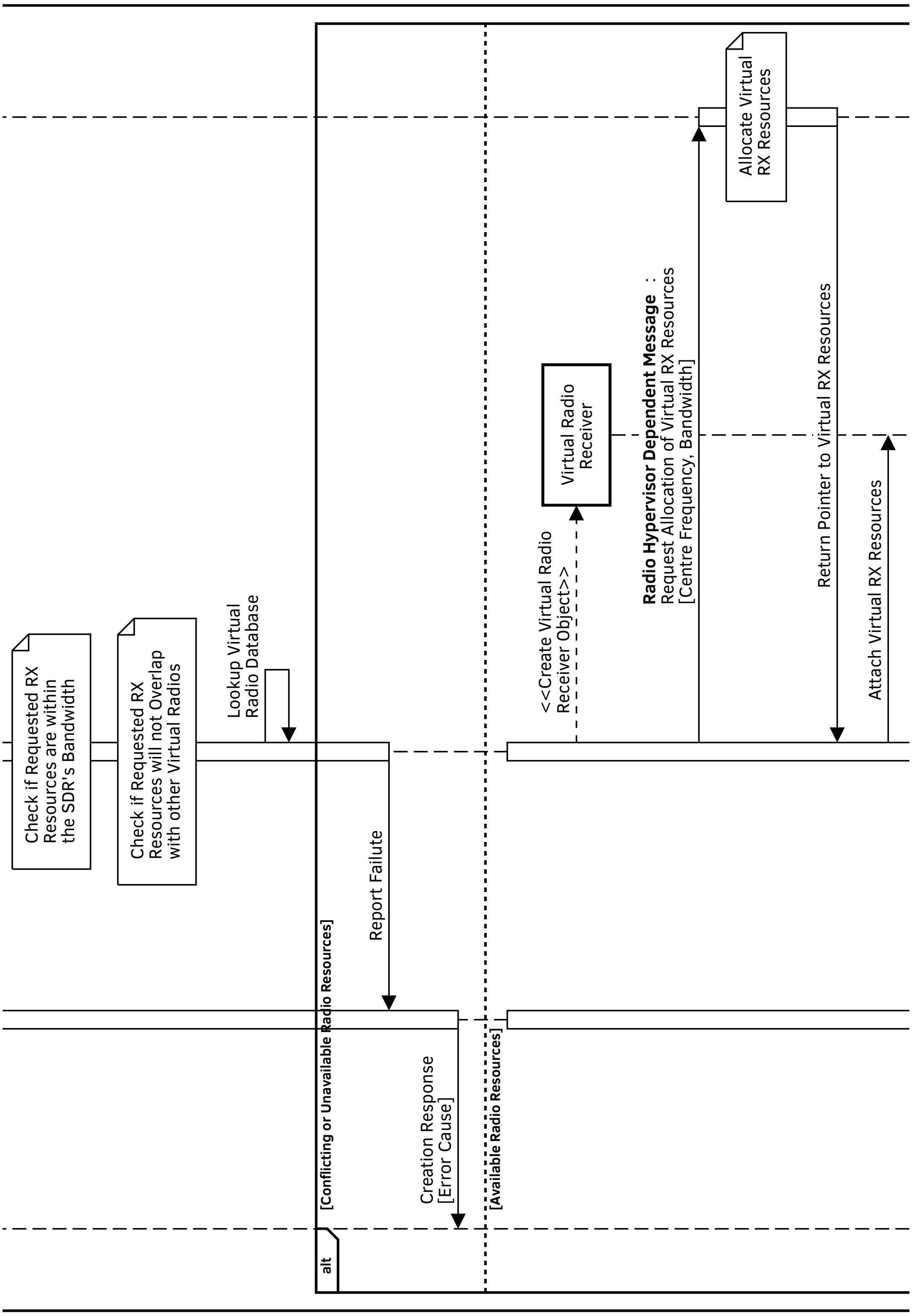
Appendix A

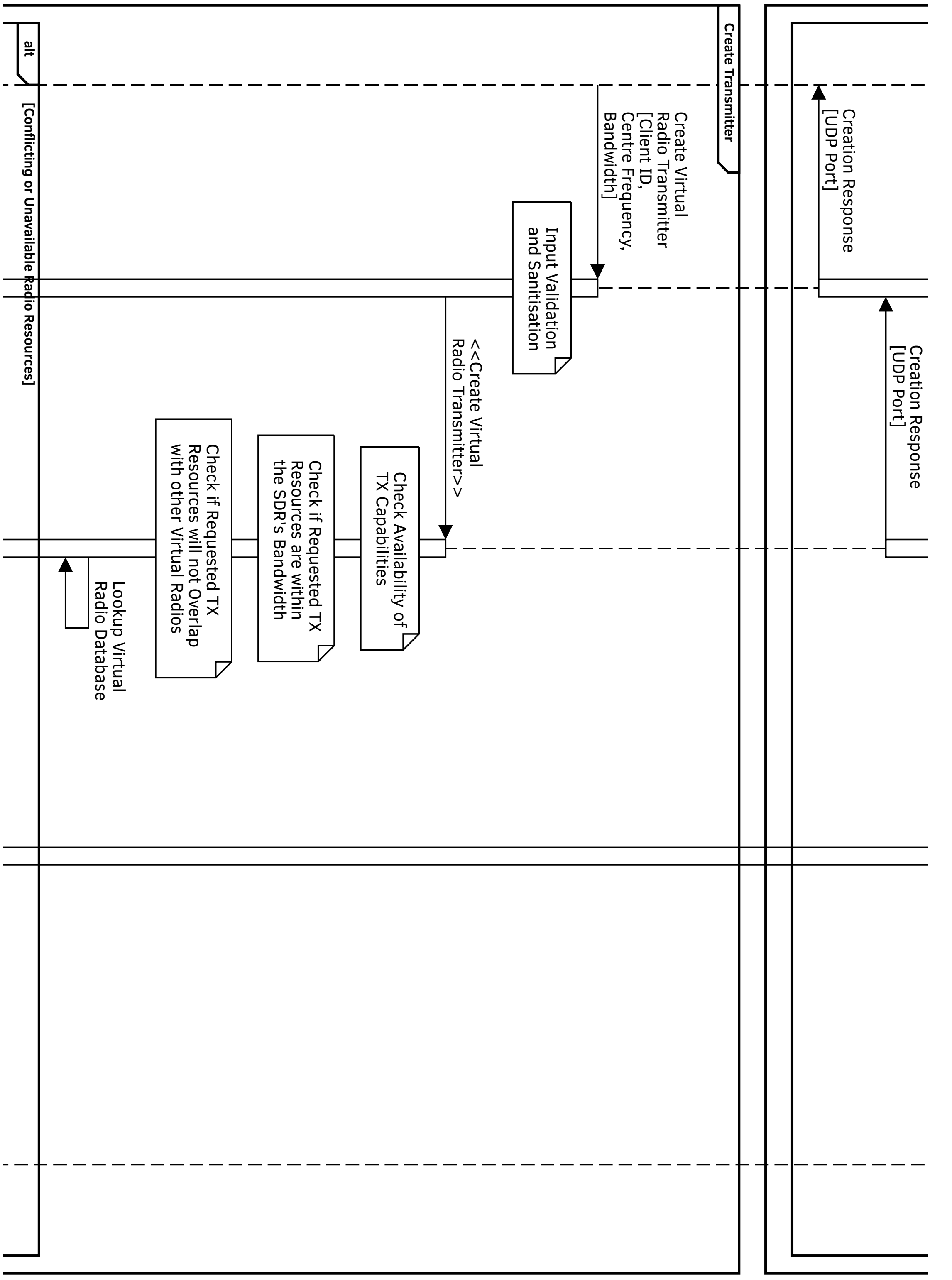
eXtensible Virtualisation Layer Interface

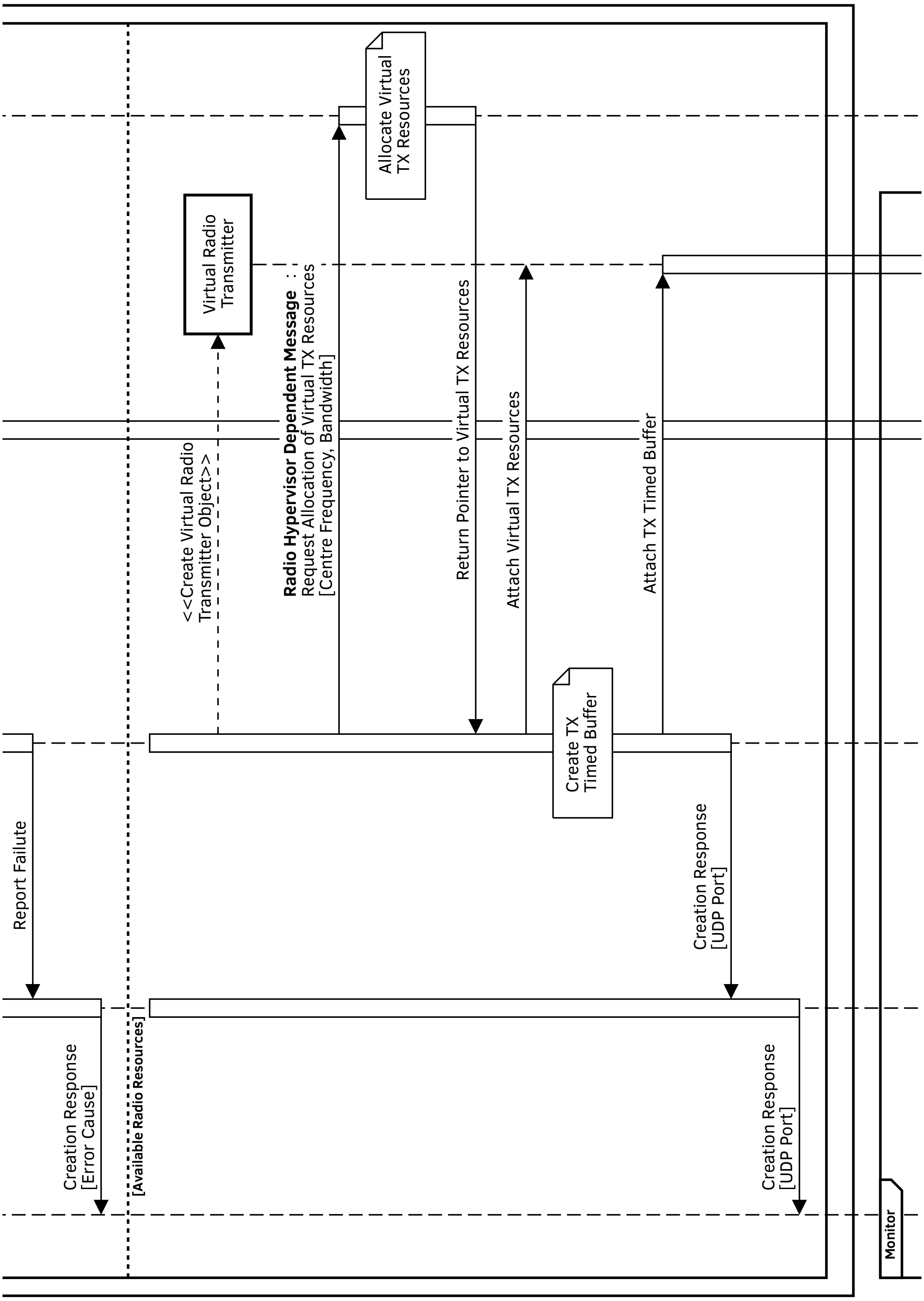
eXtensible Virtualisation Layer Interface

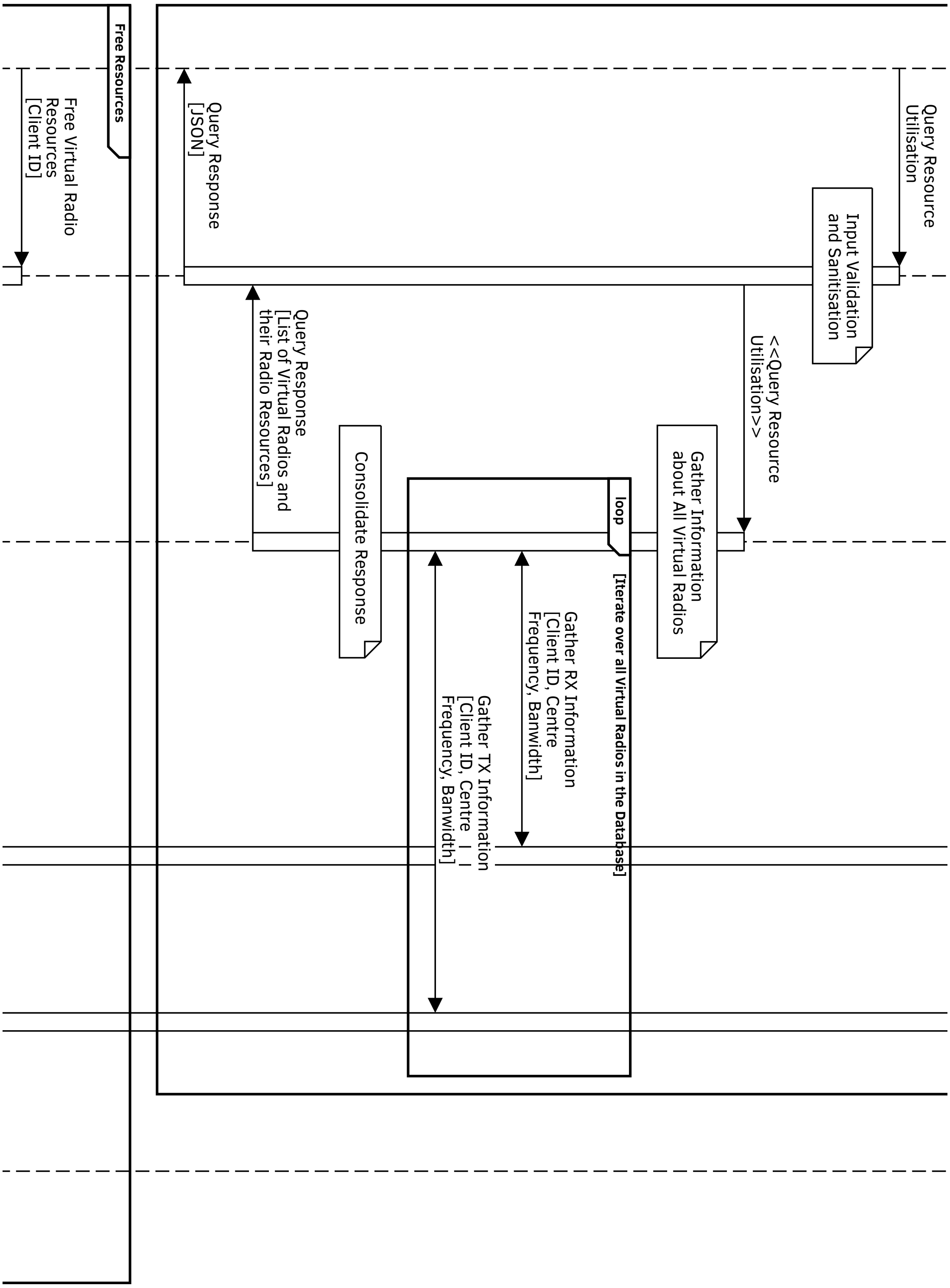
In the following, we detail the communication interfaces of the XVL introduced in Chapter 3, including both its northbound interface, used for tenants to negotiate, request, and monitor virtual radios, as well as its southbound interface, used for communicating with and controlling the underlying technology-agnostic radio hypervisor. Figure A.1 shows the sequence diagram of the exchanged messages and the interaction between the main entities of our RANaaS platform architecture.

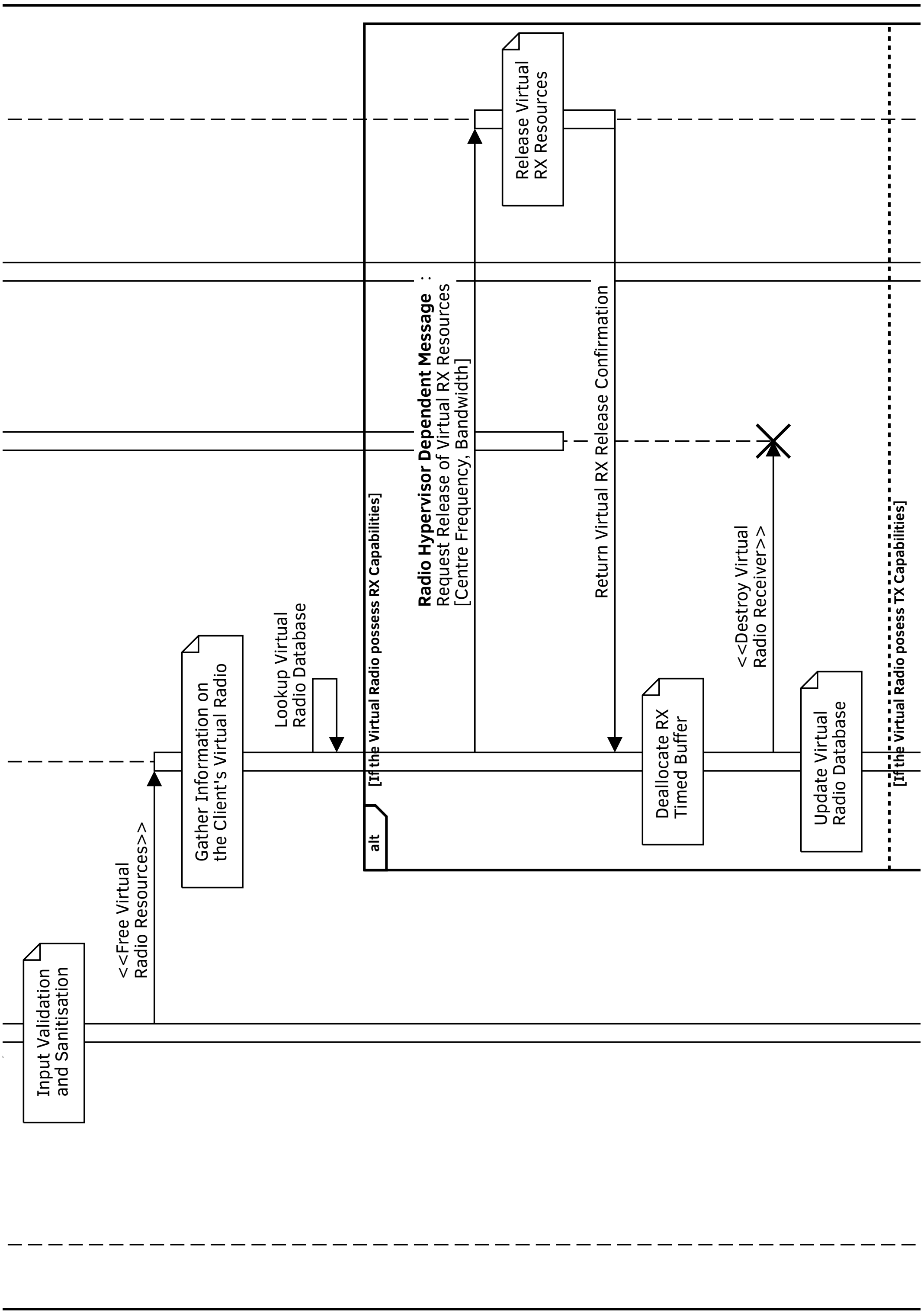

Negotiating Interface for Managing Virtual Radios
Remote Client
XVL Server
Resource Manager
Radio Hypervisor
User Input
Handshake
Synchronise Connection [Client ID]
Report Status of Server Operation
Acknowledge Connection [Server Status]
Create Receiver
Create Virtual Radio Receiver [Client ID, Centre Frequency, Bandwidth]
Input Validation and Sanitisation
<<Create Virtual Radio Receiver>>
Check Availability of RX Capabilities


Check if Requested RX Resources are within the SDR's Bandwidth
Check if Requested RX Resources will not Overlap with other Virtual Radios
Lookup Virtual Radio Database
Report Failute
[Conflicting or Unavailable Radio Resources]
alt
Creation Response [Error Cause]
[Available Radio Resources]
<<Create Virtual Radio Receiver Object>>
Virtual Radio Receiver
Radio Hypervisor Dependent Message : Request Allocation of Virtual RX Resources [Centre Frequency, Bandwidth]
Allocate Virtual RX Resources
Return Pointer to Virtual RX Resources
Attach Virtual RX Resources


Create Transmitter
alt
[Conflicting or Unavailable Radio Resources]
Create Virtual Radio Transmitter [Client ID, Centre Frequency, Bandwidth]
Creation Response [UDP Port]
Input Validation and Sanitisation
<<Create Virtual Radio Transmitter>>
Creation Response [UDP Port]
Check Availability of TX Capabilities
Check if Requested TX Resources are within the SDR's Bandwidth
Check if Requested TX Resources will not Overlap with other Virtual Radios
Lookup Virtual Radio Database


Allocate Virtual
TX Resources
Virtual Radio
Transmitter
<<Create Virtual Radio
Transmitter Object>>
Radio Hypervisor Dependent Message :
Request Allocation of Virtual TX Resources
[Centre Frequency, Bandwidth]
Return Pointer to Virtual TX Resources
Attach Virtual TX Resources
Attach TX Timed Buffer
Create TX
Timed Buffer
Report Failute
Creation Response
[UDP Port]
Creation Response
[Error Cause]
[Available Radio Resources]
Creation Response
[UDP Port]
Monitor


Free Resources
Free Virtual Radio Resources [Client ID]
Query Response [JSON]
Query Resource Utilisation
Input Validation and Sanitisation
Query Response [List of Virtual Radios and their Radio Resources]
<<Query Resource Utilisation>>
Consolidate Response
Gather Information about All Virtual Radios
loop
[Iterate over all Virtual Radios in the Database]
Gather RX Information [Client ID, Centre Frequency, Banwidth]
Gather TX Information [Client ID, Centre Frequency, Banwidth]


Input Validation
and Sanitisation
<<Free Virtual
Radio Resources>>
Gather Information on
the Client's Virtual Radio
Lookup Virtual
Radio Database
alt
[If the Virtual Radio possess RX Capabilities]
Radio Hypervisor Dependent Message :
Request Release of Virtual RX Resources
[Centre Frequency, Bandwidth]
Release Virtual
RX Resources
Return Virtual RX Release Confirmation
Deallocate RX
Timed Buffer
<<Destroy Virtual
Radio Receiver>>
Update Virtual
Radio Database
[If the Virtual Radio posess TX Capabilities]

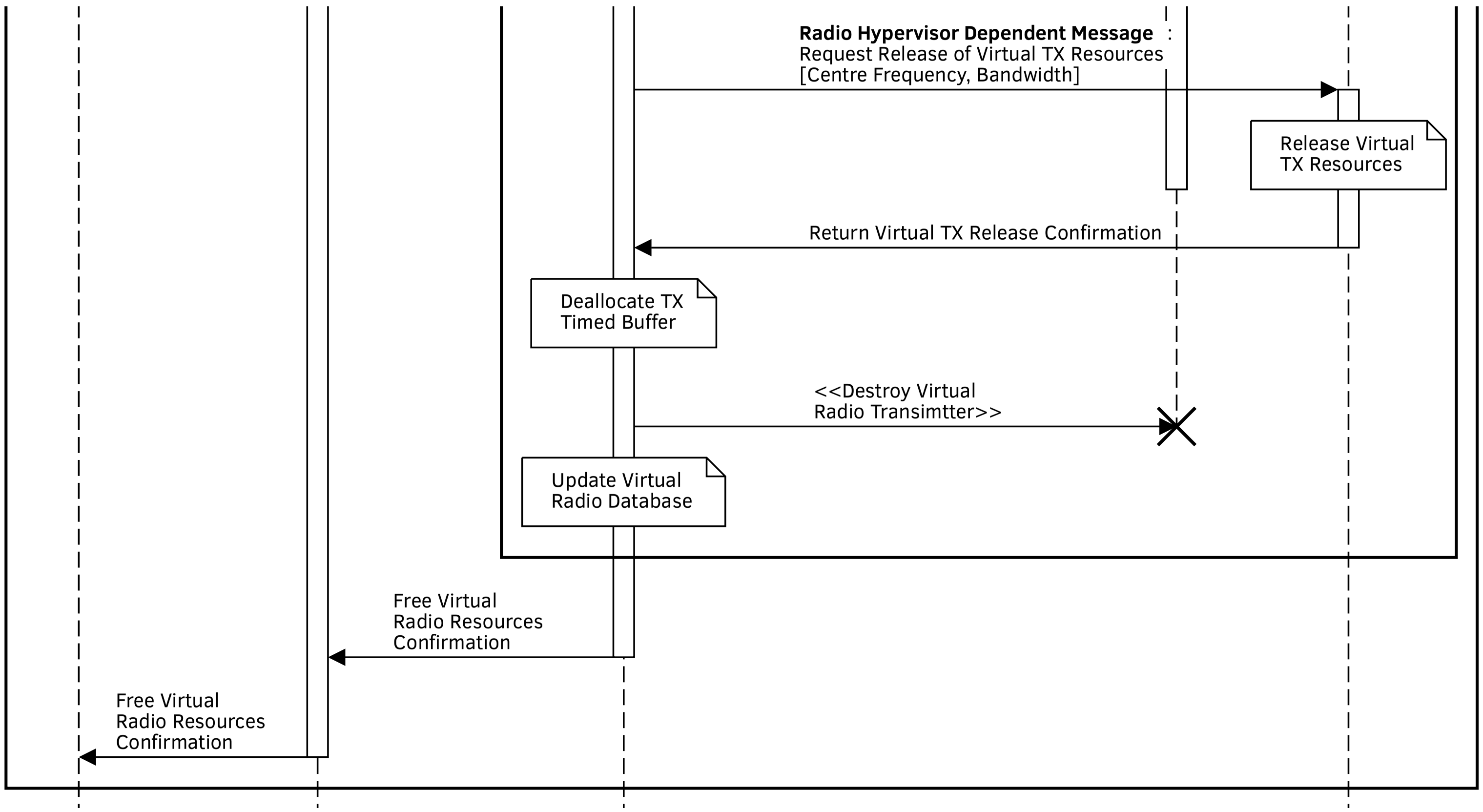


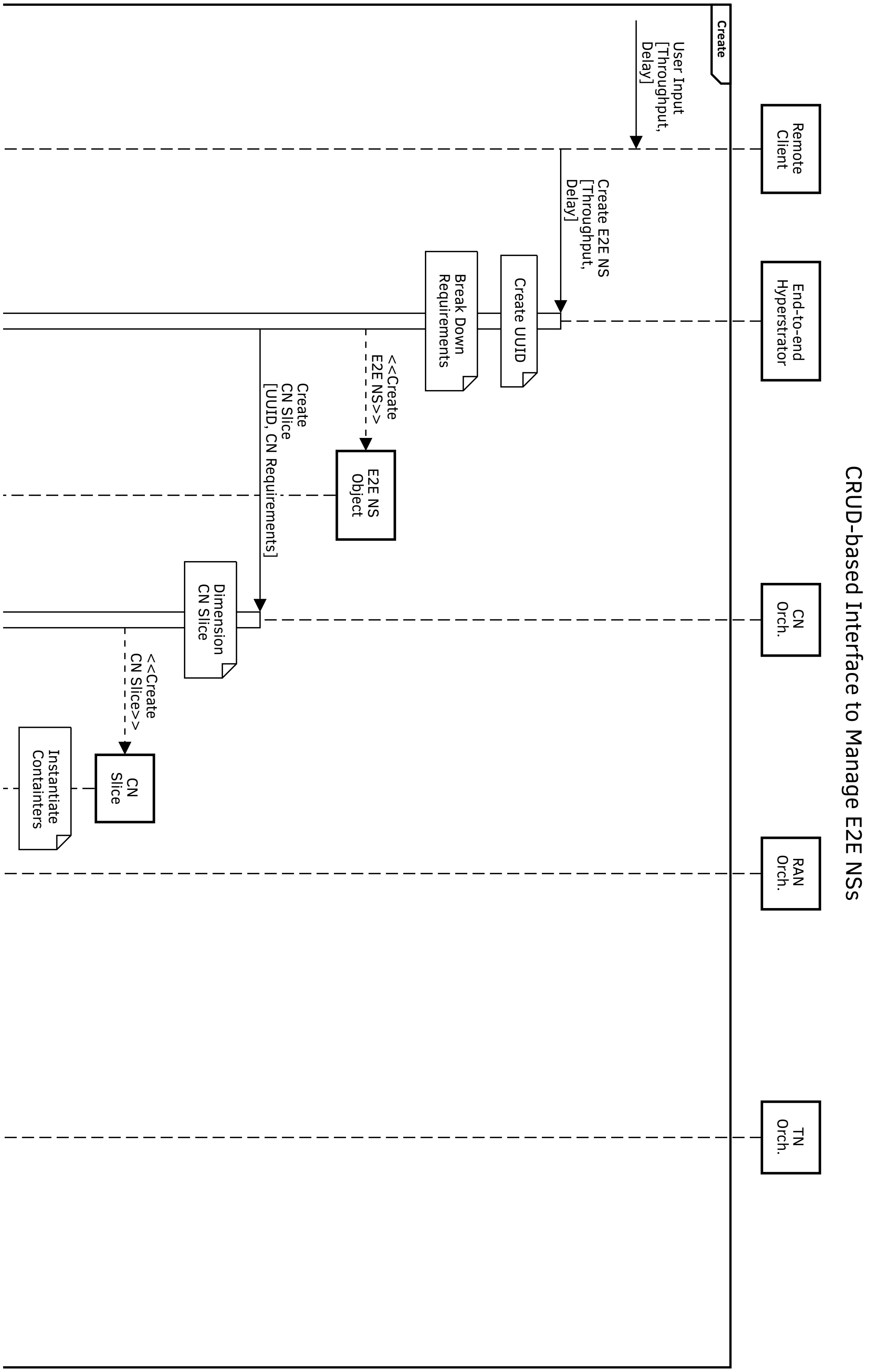
FIGURE A.1: The sequence diagram showing the operations that tenants can perform over the virtual radio resources through XVL's northbound interface, including creating different types of virtual radio transmitters/receivers and monitoring the allocation of resources; and the interactions with an underlying technology-agnostic radio hypervisor for carrying out resource management operations through its southbound interface.

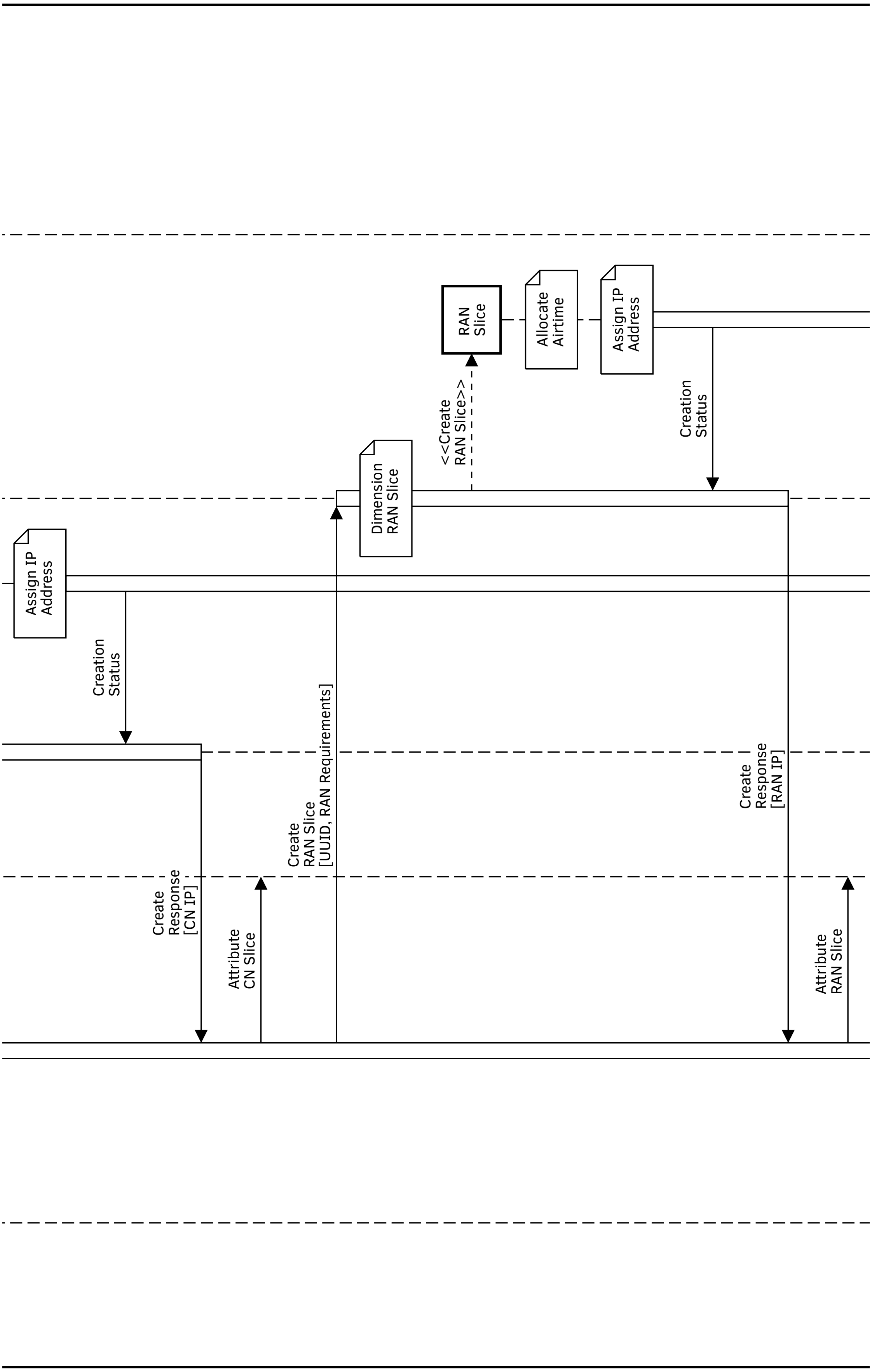
Appendix B

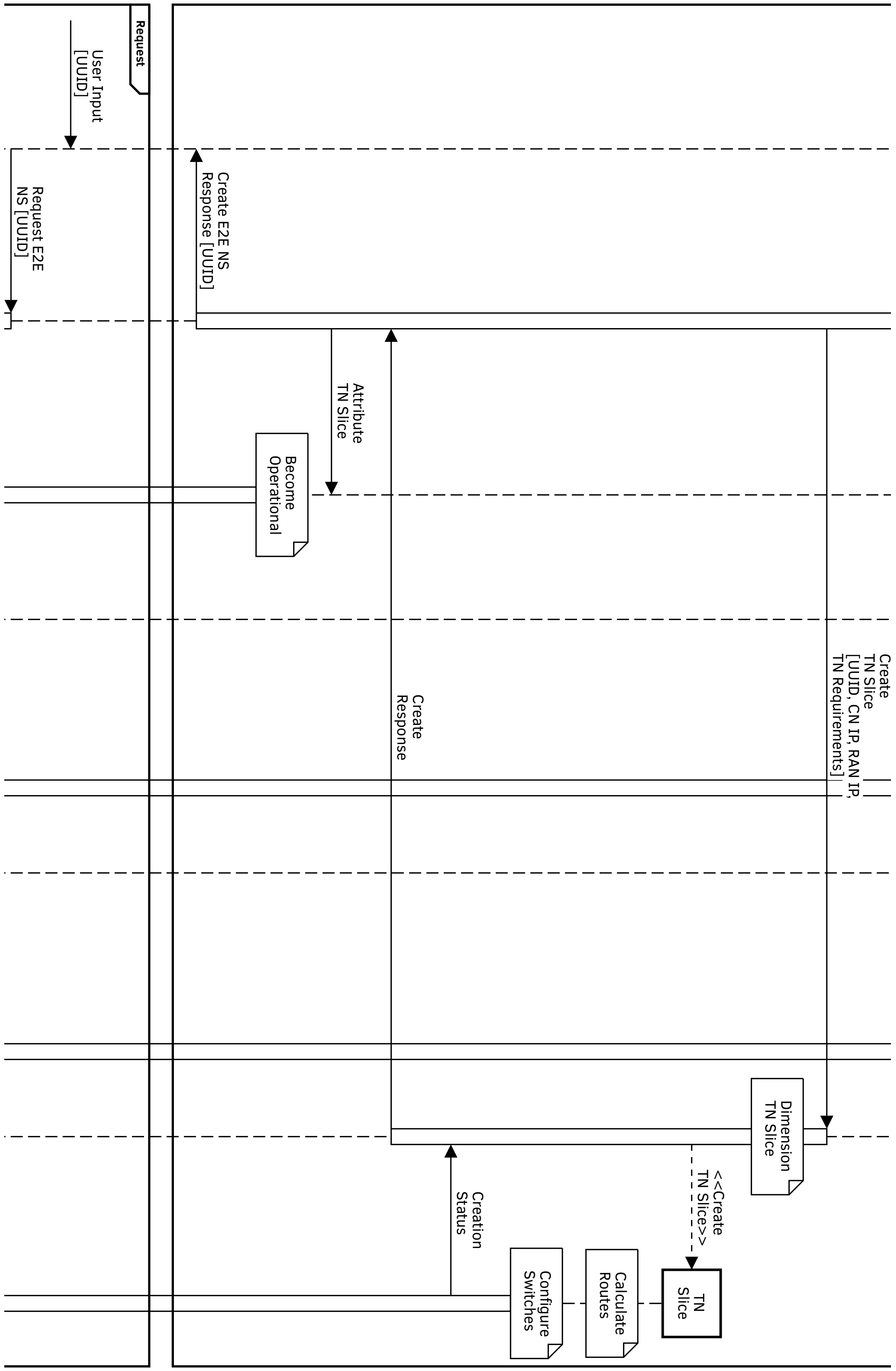
End-to-End Hyperstrator Interface

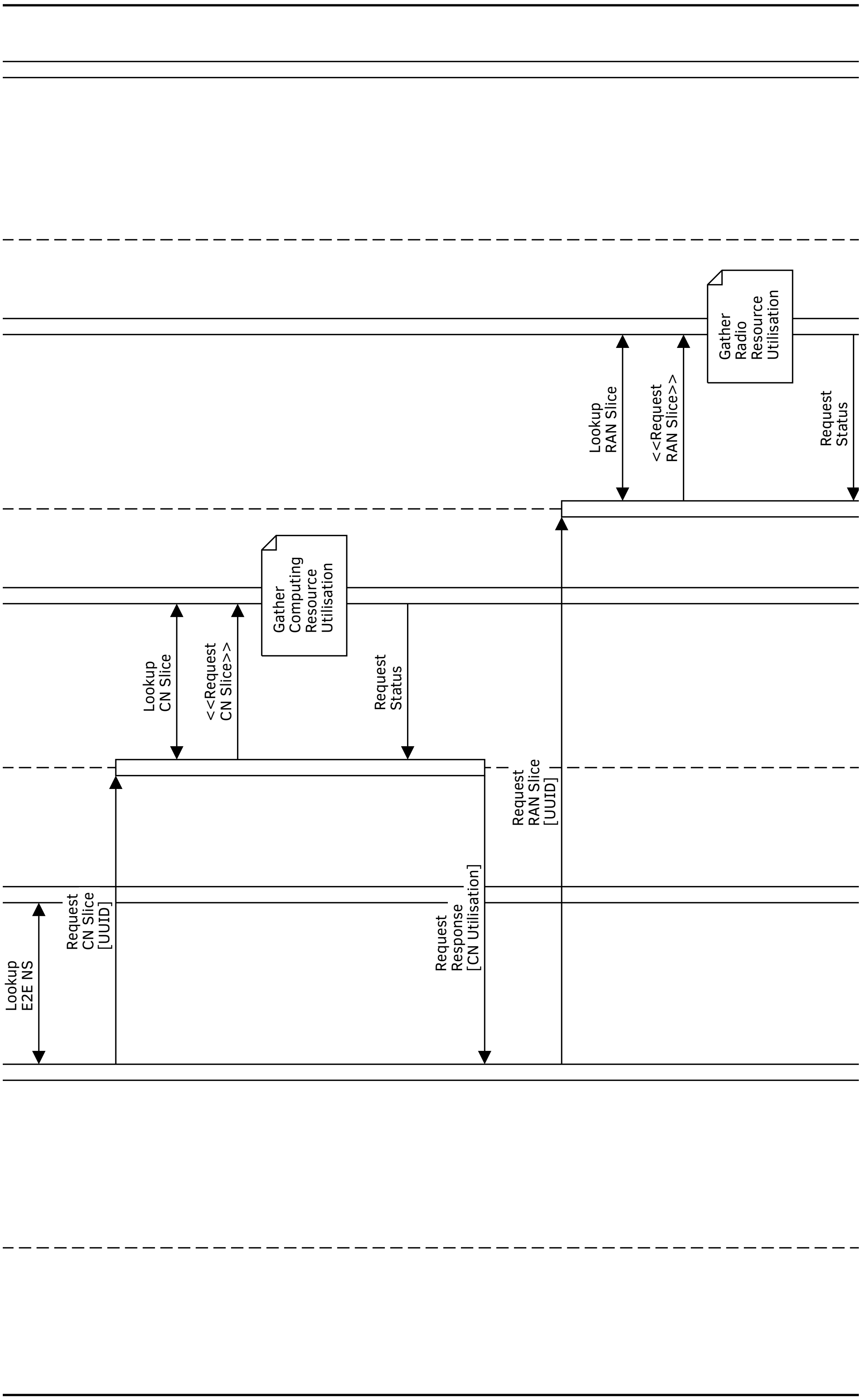
End-to-End Hyperstrator Interface

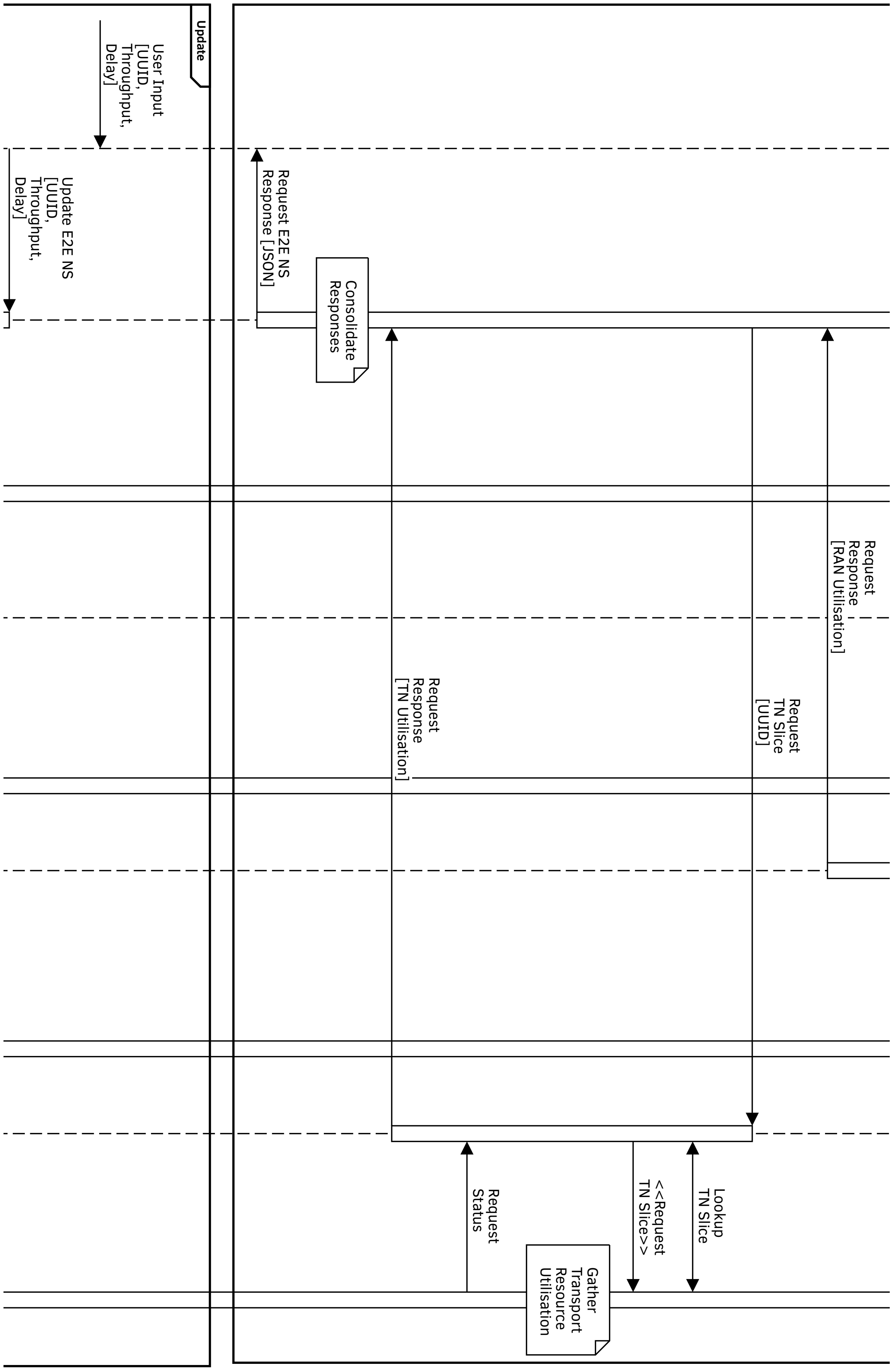
In the following, we detail the communication interfaces of the hyperstrator introduced in Chapter 5, including both its northbound interface, used for tenants to operate over E2E NSs, and its southbound interface, used to coordinate the resource allocation across the distributed specialised orchestrators. Figure B.1 shows the sequence diagram of the exchanged messages and the interaction between the main entities of our hierarchical orchestration architecture for E2E communication networks.

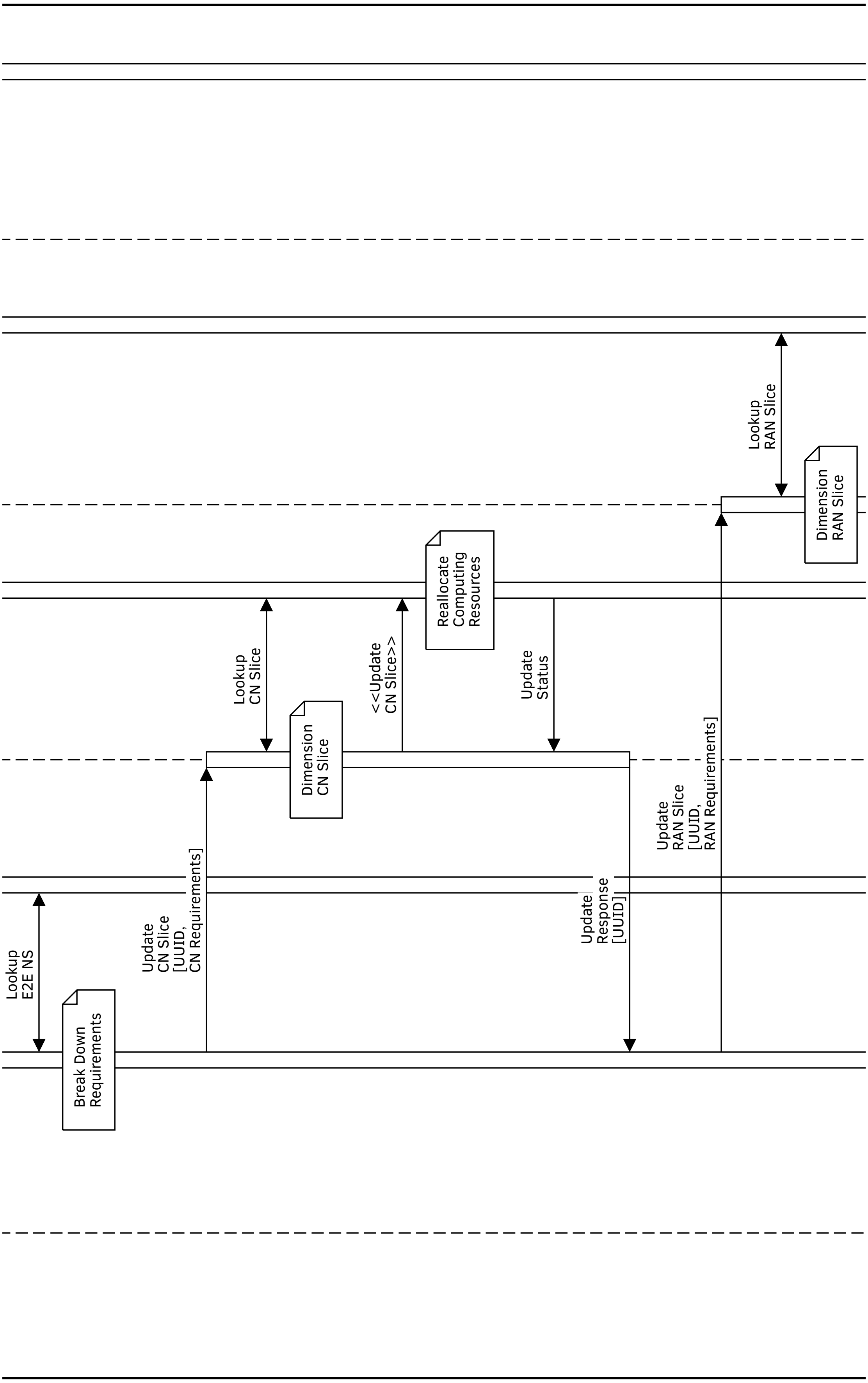

CRUD-based Interface to Manage E2E NSs
Create
Remote Client
End-to-end Hyperstrator
CN Orch.
RAN Orch.
TN Orch.
User Input [Throughput, Delay]
Create E2E NS [Throughput, Delay]
Create UUID
Break Down Requirements
<<Create E2E NS>>
E2E NS Object
Create CN Slice [UUID, CN Requirements]
Dimension CN Slice
<<Create CN Slice>>
CN Slice
Instantiate Containters

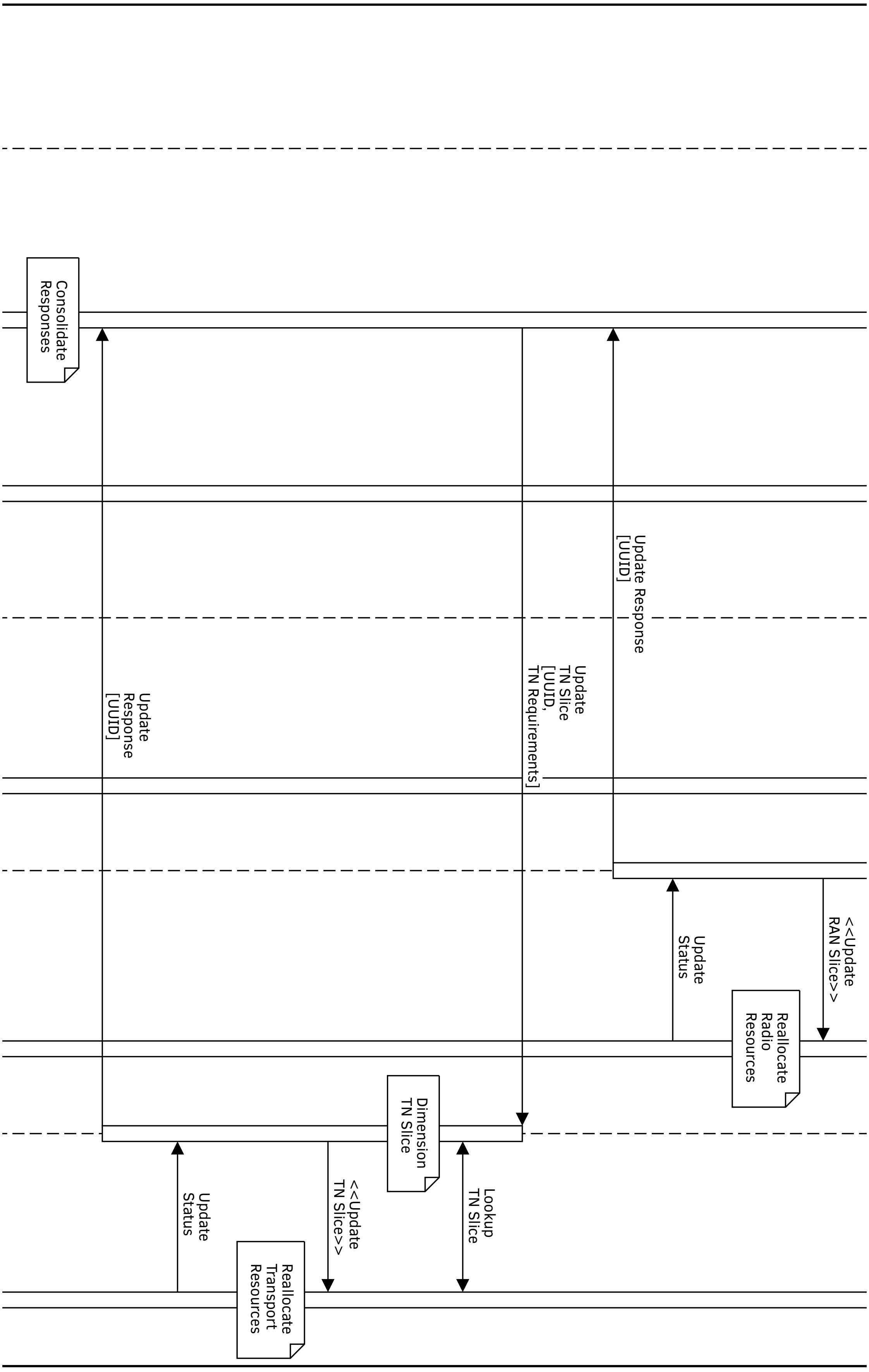

RAN Slice
Allocate Airtime
Assign IP Address
Creation Status
<<Create RAN Slice>>
Dimension RAN Slice
Assign IP Address
Creation Status
Create RAN Slice [UUID, RAN Requirements]
Create Response [RAN IP]
Create Response [CN IP]
Attribute CN Slice
Attribute RAN Slice

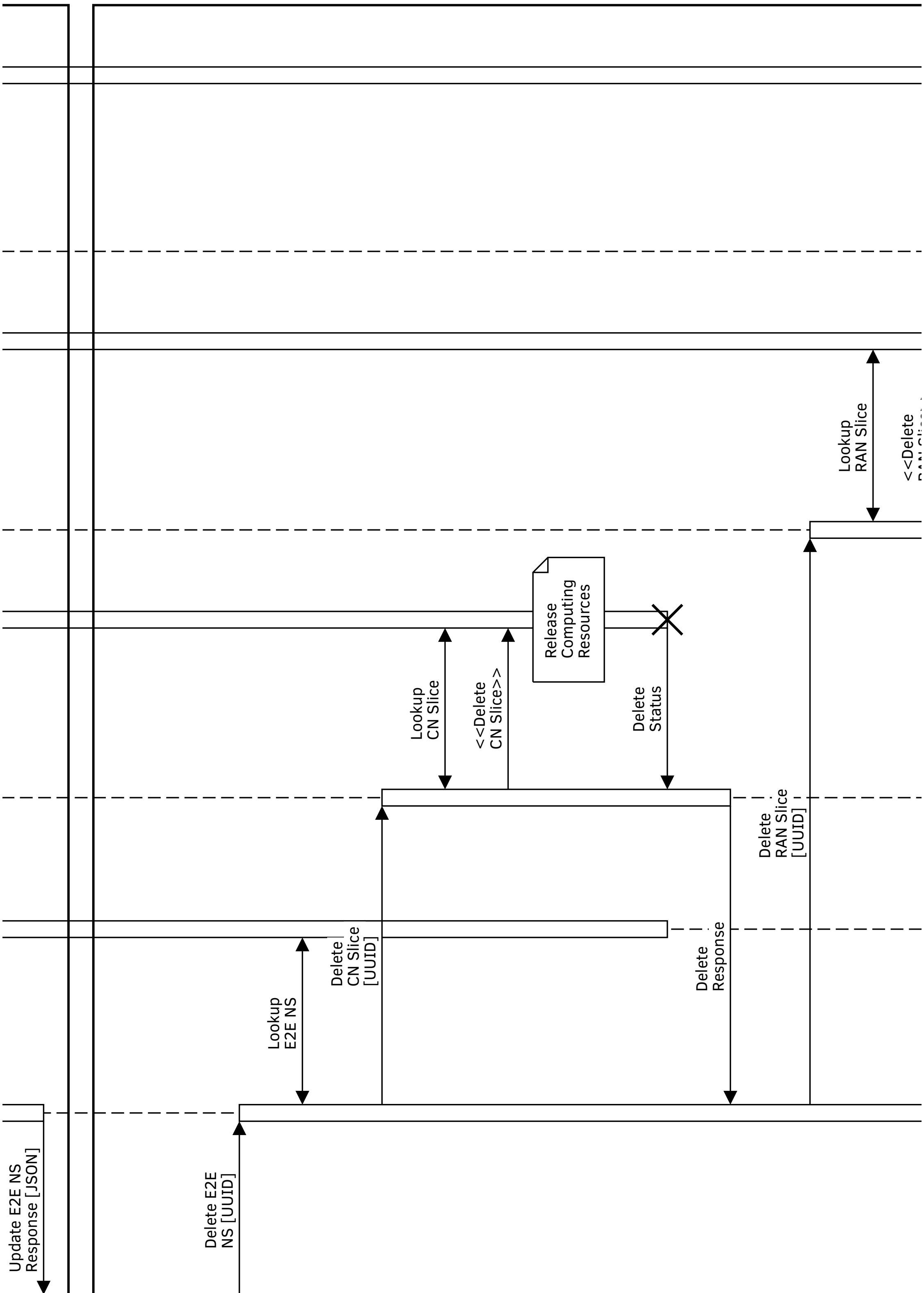

Request
User Input [UUID]
Request E2E NS [UUID]
Create E2E NS Response [UUID]
Attribute TN Slice
Become Operational
Create Response
Create TN Slice [UUID, CN IP, RAN IP, TN Requirements]
Dimension TN Slice
<<Create TN Slice>>
Creation Status
Configure Switches
Calculate Routes
TN Slice


Lookup
E2E NS
Request
CN Slice
[UUID]
Lookup
CN Slice
<<Request
CN Slice>>
Gather
Computing
Resource
Utilisation
Request
Status
Request
Response
[CN Utilisation]
Request
RAN Slice
[UUID]
Lookup
RAN Slice
<<Request
RAN Slice>>
Gather
Radio
Resource
Utilisation
Request
Status


Update
User Input [UUID, Throughput, Delay]
Update E2E NS [UUID, Throughput, Delay]
Request E2E NS Response [JSON]
Consolidate Responses
Request Response [RAN Utilisation]
Request Response [TN Utilisation]
Request TN Slice [UUID]
Request Status
<<Request TN Slice>>
Lookup TN Slice
Gather Transport Resource Utilisation


Lookup E2E NS
Break Down Requirements
Update CN Slice [UUID, CN Requirements]
Lookup CN Slice
Dimension CN Slice
<<Update CN Slice>>
Reallocate Computing Resources
Update Status
Update Response [UUID]
Update RAN Slice [UUID, RAN Requirements]
Lookup RAN Slice
Dimension RAN Slice


Lookup
TN Slice
Dimension
TN Slice
<<Update
TN Slice>>
Reallocate
Transport
Resources
Update
Status
Update
Response
[UUID]
Update
TN Slice
[UUID,
TN Requirements]
<<Update
RAN Slice>>
Reallocate
Radio
Resources
Update
Status
Update Response
[UUID]
Consolidate
Responses


Delete
User Input [UUID]
Update E2E NS Response [JSON]
Delete E2E NS [UUID]
Lookup E2E NS
Delete CN Slice [UUID]
Lookup CN Slice
<<Delete CN Slice>>
Release Computing Resources
Delete Status
Delete Response
Delete RAN Slice [UUID]
Lookup RAN Slice
<<Delete

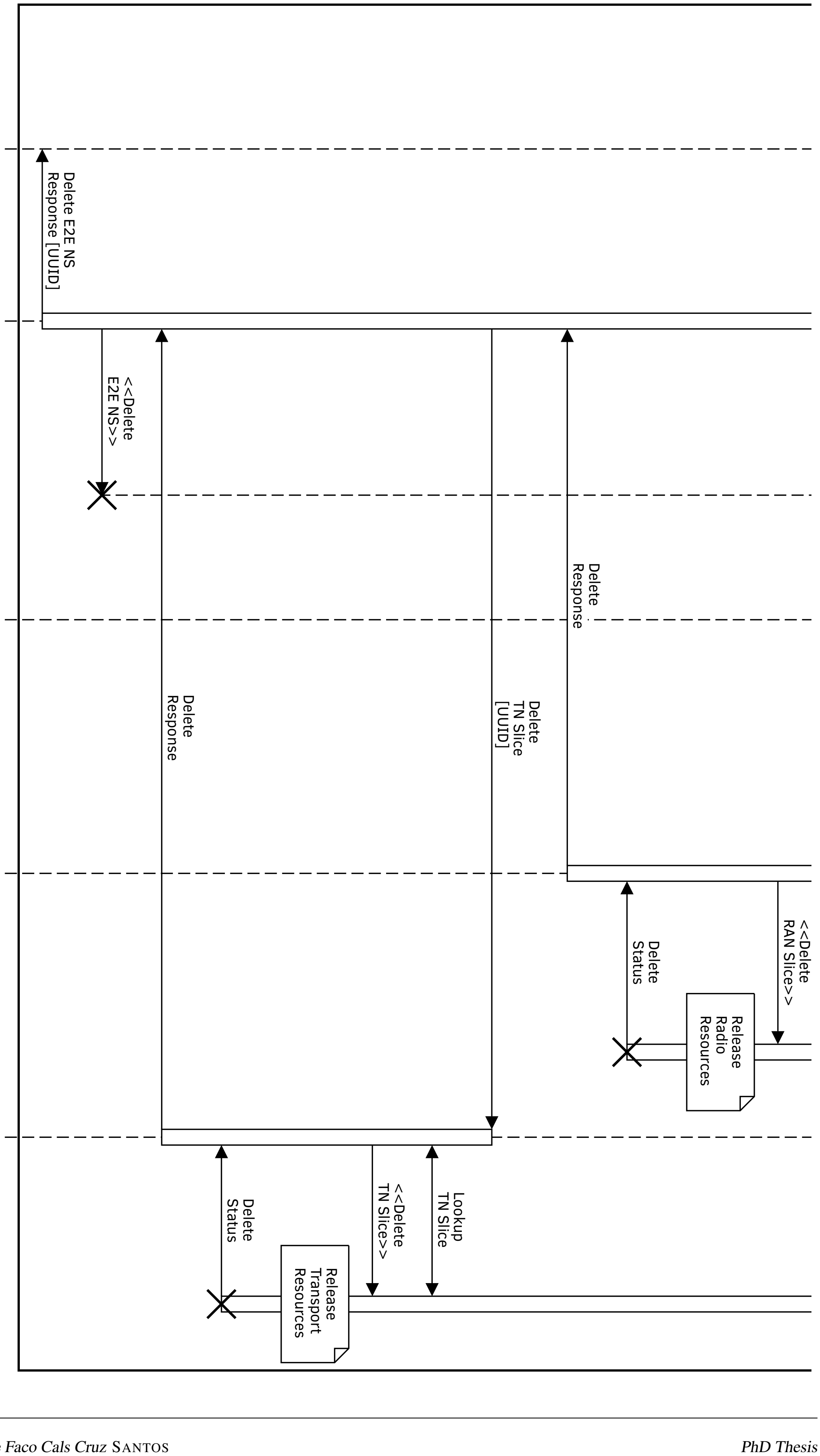


FIGURE B.1: The sequence diagram showing the operations that tenants can perform over the E2E NSs through the hyperstrator's northbound interface; and the coordination between multiple orchestrators over the E2E NSs through the hyperstrator's southbound interface.

Acronyms

3GPP	3rd Generation Partnership Project
ACL	Access Control List
ANN	Artificial Neural Network
ARM	Advanced RISC Machines
ARPU	Average Revenue Per User
ASIC	Application-Specific Integrated Circuit
B2B	Business-to-business
B2B2C	Business-to-business-to-consumer
B2C	Business-to-consumer
BBU	Baseband Unit
C-RAN	Centralised-RAN
CAGR	Compound Annual Growth Rate
CBRS	Citizens Broadband Radio Service
CDN	Content Delivery Network
CN	Core Network
CNaaS	CN as a Service
COTS	Commercial-Off-The-Shelf
CP	Cloud Provider
CriC	Critical Communications
CRUD	Create, Read, Update, Delete
D-RAN	Distributed-RAN
DCN	Data Centre Network
DSP	Digital Signal Processor
E2E	End-to-End

eMBB	Enhanced Mobile Broadband
EPC	Evolved Packet Core
eV2X	Enhanced Vehicular-to-Everything
FOTSP	Freely Open-loop TSP
FPGA	Field-Programmable Gate Array
GA	Greedy Algorithm
GEFI	Global Experimentation for Future Internet
GPP	General-purpose Processor
I/O	Input/Output
IoT	Internet of Things
IQ	In-phase & Quadrature
ISP	Internet Service Provider
KPI	Key Performance Indicator
LVM	Logical Volume Manager
LXC	Linux Containers
MANO	Management and Orchestration
MBB	Mobile Broadband
MBMS	Multimedia Broadcast Multicast Service
MILP	Mixed-Integer Linear Programming
MIoT	Massive IoT
MNO	Mobile Network Operator
MSE	Mean Square Error
MVNO	Mobile Virtual Network Operator
NB-IoT	Narrowband IoT
NFV	Network Function Virtualisation
NFVO	NFV Orchestrator
NFVO-C	NFVO Composite
NFVO-N	NFVO Nested
NNA	Nearest Neighbour Algorithm
NS	Network Slice
NSaaS	NS as a Service

NSS	Network Segment Slice
ONOS	Open Network Operating System
OOBE	Out-of-band Emission
ORCA	Orchestration and Reconfiguration Control Architecture
OS	Operating System
OSM	Open-source MANO
OSS	Operation Support System
PNF	Physical Network Function
PRB	Physical Resource Block
QoS	Quality of Service
RAN	Radio Access Network
RANaaS	RAN as a Service
RAT	Radio Access Technology
RF	Radio Frequency
RNC	Radio Network Controller
RQ	Research Question
RRH	Remote Radio Head
SDK	Software Development Kit
SDN	Software-defined Networking
SDR	Software-defined Radio
SDS	Software-defined Switch
SIR	Signal-to-Interference Ratio
SLA	Service Level Agreement
SON	Self Organising Network
TN	Transport Network
TSP	Travelling Salesman Problem
U-NII	Unlicensed National Information Infrastructure
UE	User Equipment
UUID	Universally Unique Identifier
V2X	Vehicular-to-Everything
VM	Virtual Machine

VNF	Virtual Network Function
VPN	Virtual Private Network
VRB	Virtual Resource Block
VRF	Virtual Radio Function
XaaS	Everything as a Service
XVL	eXtensible Virtualisation Layer

Bibliography

[1] A. Checko, H. L. Christiansen, Y. Yan, *et al.*, "Cloud RAN for Mobile Networks—A Technology Overview", *IEEE Communications Surveys & Tutorials (COMST)*, vol. 17, no. 1, pp. 405–426, 2014.

[2] B. G. Mölleryd, J. Markendahl, and Ö. Mäkitalo, "Analysis of Operator Options to Reduce the Impact of the Revenue Gap Caused by Flat Rate Mobile Broadband Subscriptions", in *Conference on Telecom, Media & Internet Tele-Economics (CTTE)*, 2009.

[3] J. Markendahl, Ö. Mäkitalo, J. Werding, and B. G. Mölleryd, "Business Innovation Strategies to Reduce the Revenue Gap for Wireless Broadband Services", *IDATE Communications & Strategies (CommStrat)*, no. 75, p. 35, 2009.

[4] L. Doyle, J. Kibiłda, T. K. Forde, and L. DaSilva, "Spectrum without Bounds, Networks without Borders", *Proceedings of the IEEE (PIEEE)*, vol. 102, no. 3, pp. 351–365, 2014.

[5] P. Rost, C. Mannweiler, D. S. Michalopoulos, *et al.*, "Network Slicing to Enable Scalability and Flexibility in 5G Mobile Networks", *IEEE Communications magazine (ComMag)*, vol. 55, no. 5, pp. 72–79, 2017.

[6] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network Slicing in 5G: Survey and Challenges", *IEEE Communications Magazine (ComMag)*, vol. 55, no. 5, pp. 94–100, 2017.

[7] 3rd Generation Partnership Project, "3GPP TR 22.891: Feasibility Study on New Services and Markets Technology Enablers", 3rd Generation Partnership Project, Tech. Rep., version 14.2.0, Sep. 2016.

[8] 5G Americas, "Network Slicing for 5G Networks & Services", 5G Americas, Tech. Rep., version 11.21, Nov. 2016. [Online]. Available: `https://www.5gamericas.org/wp-content/uploads/2019/07/5G_Americas_Network_Slicing_11.21_Final.pdf`.

[9] J. Ordonez-Lucena, P. Ameigeiras, D. Lopez, *et al.*, “Network Slicing for 5G with SDN/NFV: Concepts, Architectures, and Challenges”, *IEEE Communications Magazine (ComMag)*, vol. 55, no. 5, pp. 80–87, 2017.

[10] 3rd Generation Partnership Project, “3GPP TR 28.801: Study on Management and Orchestration of Network Slicing for Next Generation Network”, 3rd Generation Partnership Project, Tech. Rep., version 1.2.0, May 2017.

[11] ——, “3GPP TR 38.801: Study on New Radio Access Technology: Radio Access Architecture and Interfaces”, 3rd Generation Partnership Project, Tech. Rep., version 14.0.0, Mar. 2017.

[12] C. Sexton, N. J. Kaminski, J. M. Marquez-Barja, N. Marchetti, and L. A. DaSilva, “5G: Adaptable Networks Enabled by Versatile Radio Access Technologies”, *IEEE Communications Surveys & Tutorials (COMST)*, vol. 19, no. 2, pp. 688–720, 2017.

[13] M. Jiang, M. Condoluci, and T. Mahmoodi, “Network Slicing in 5G: An Auction-based Model”, in *IEEE International Conference on Communications (ICC)*, 2017, pp. 1–6.

[14] K. Samdanis, X. Costa-Perez, and V. Sciancalepore, “From Network Sharing to Multitenancy: The 5G Network Slice Broker”, *IEEE Communications Magazine (ComMag)*, vol. 54, no. 7, pp. 32–39, 2016.

[15] T. K. Forde, I. Macaluso, and L. E. Doyle, “Exclusive Sharing & Virtualization of the Cellular Network”, in *IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2011, pp. 337–348.

[16] X. Zhou, R. Li, T. Chen, and H. Zhang, “Network Slicing as a Service: Enabling Enterprises’ Own Software-Defined Cellular Networks”, *IEEE Communications Magazine (ComMag)*, vol. 54, no. 7, pp. 146–153, 2016.

[17] 3rd Generation Partnership Project, “3GPP TR 28.530: Management and Orchestration; Concepts, Use Cases and Requirements”, 3rd Generation Partnership Project, Tech. Rep., version 15.1.0, Dec. 2018.

[18] T. Taleb, M. Corici, C. Parada, *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.

[19] D. Sabella, P. Rost, Y. Sheng, *et al.*, “RAN as a Service: Challenges of Designing a Flexible RAN Architecture in a Cloud-Based Heterogeneous Mobile Network”, *IEEE Future Network & Mobile Summit (FUNEMS)*, 2013.

[20] R. Sherwood, G. Gibb, K.-K. Yap, *et al.*, "FlowVisor: A Network Virtualization Layer", *OpenFlow Switch Consortium, Tech. Rep*, vol. 1, p. 132, 2009.

[21] M. Kist, J. Rochol, L. A. Dasilva, and C. B. Both, "SDR Virtualization in Future Mobile Networks : Enabling Multi-Programmable Air-Interfaces", in *IEEE International Conference on Communications (ICC)*, May 2018.

[22] J. van de Belt, L. Doyle, *et al.*, "Defining and Surveying Wireless Link and Network Virtualization", *IEEE Communications Surveys & Tutorials (COMST)*, vol. 19, no. 3, pp. 1603–1627, 2017.

[23] M. M. Saad, F. A. Bhatti, A. Zafar, *et al.*, "Air-interface Virtualization Using Filter Bank Multicarrier and Orthogonal Frequency Division Multiplexing Configurations", *Wiley Transactions on Emerging Telecommunications Technologies (ETT)*, e4154, 2020.

[24] J. Sachs and S. Baucke, "Virtual Radio: A Framework for Configurable Radio Networks", in *ACM Conference on Wireless Internet (WICON)*, 2008, pp. 1–7.

[25] M. Howard, "The Evolution of SDN and NFV Orchestration", Infonetics Research, Tech. Rep., Sep. 2015. [Online]. Available: `https://investor.juniper.net/files/doc_downloads/2015/May/Latest%20Resources/2000604-en_v001_h27j2q.pdf`.

[26] H. Flinck, C. Satorti, A. Andrianov, C. Mannweiler, and N. Sprecher, "Network Slicing Management and Orchestration", Internet Engineering Task Force, Tech. Rep., 2017.

[27] 5G PPP Architecture Working Group. (Sep. 2014). View on 5G Architecture, [Online]. Available: `https://5g-ppp.eu/wp-content/uploads/2014/02/5G-PPP-5G-Architecture-WP-July-2016.pdf`.

[28] P. Berde, M. Gerola, J. Hart, *et al.*, "ONOS: Towards An Open, Distributed SDN OS", in *ACM Workshop on Hot Topics in Software Defined Networks (HotSDN)*, 2014, pp. 1–6.

[29] European Telecommunications Standards Institute. (Dec. 2020). Open Source MANO, [Online]. Available: `https://osm.etsi.org/`.

[30] L. Peterson, A. Al-Shabibi, T. Anshutz, *et al.*, "Central Office Re-Architected as a Data Center", *IEEE Communications Magazine (ComMag)*, vol. 54, no. 10, pp. 96–101, 2016.

[31] R. Riggio, M. K. Marina, J. Schulz-Zander, S. Kuklinski, and T. Rasheed, "Programming Abstractions for Software-defined Wireless Networks", *IEEE Transactions on Network and Service Management (TNSM)*, vol. 12, no. 2, pp. 146–162, 2015.

[32] K. Katsalis, N. Nikaein, and A. Edmonds, “Multi-domain Orchestration for NFV: Challenges and Research Directions”, in *IEEE International Conference on Ubiquitous Computing and Communications and International Symposium on Cyberspace and Security (IUCC-CSS)*, 2016, pp. 189–195.

[33] A. S. Tanenbaum and T. Austin, *Structured Computer Organization*. Pearson, 2014.

[34] W. Liu, J. F. Santos, J. v. de Belt, *et al.*, “Enabling Virtual Radio Functions on Software Defined Radio for Future Wireless Networks”, *Springer Wireless Personal Communications (WPC)*, pp. 1–17, 2020.

[35] M. Joler, “How FPGAs can Help Create Self-Recoverable Antenna Arrays”, *Hindawi International Journal of Antennas and Propagation (IJAP)*, vol. 2012, 2012.

[36] R. Pellizzoni and M. Caccamo, “Hybrid Hardware-software Architecture for Reconfigurable Real-time Systems”, in *IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2008, pp. 273–284.

[37] H. Marcondes and A. A. Fröhlich, “A Hybrid Hardware and Software Component Architecture for Embedded System Design”, in *Springer International Embedded Systems Symposium (IESS)*, 2009, pp. 259–270.

[38] J. Santiago da Silva, F.-R. Boyer, and J. P. Langlois, “P4-compatible High-level Synthesis of Low Latency 100 Gb/s Streaming Packet Parsers in FPGAs”, in *ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA)*, 2018, pp. 147–152.

[39] P. Di Francesco, S. McGettrick, U. K. Anyanwu, *et al.*, “A Split MAC Approach for SDR Platforms”, *IEEE Transactions on Computers (TC)*, vol. 64, no. 4, pp. 912–924, 2014.

[40] B. Moons and M. Verhelst, “An Energy-efficient Precision-scalable ConvNet Processor in 40-nm CMOS”, *IEEE Journal of Solid-state Circuits (JSSC)*, vol. 52, no. 4, pp. 903–914, 2016.

[41] I. Moerman, D. Zeghlache, A. Shahid, *et al.*, “Mandate-driven Networking Eco-system: A Paradigm Shift in End-to-End Communications”, in *6G Wireless Summit (6G SUMMIT)*, 2020, pp. 1–6.

[42] M. Condoluci and T. Mahmoodi, “Softwarization and Virtualization in 5G Mobile Networks: Benefits, Trends and Challenges”, *Elsevier Computer Networks (ComNet)*, vol. 146, pp. 65–84, 2018.

[43] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, “Network Slicing and Softwarization: A survey on Principles, Enabling Technologies, and Solutions”, *IEEE Communications Surveys & Tutorials (COMST)*, vol. 20, no. 3, pp. 2429–2453, 2018.

[44] H.-H. Cho, C.-F. Lai, T. K. Shih, and H.-C. Chao, “Integration of SDR and SDN for 5G”, *IEEE Access*, vol. 2, 1196–:w 1204, 2014.

[45] N. McKeown, T. Anderson, H. Balakrishnan, *et al.*, “OpenFlow: Enabling Innovation in Campus Networks”, *ACM SIGCOMM Computer Communication Review (CCR)*, vol. 38, no. 2, pp. 69–74, 2008.

[46] A. Singh, J. Ong, A. Agarwal, *et al.*, “Jupiter Rising: A Fecade of Clos topologies and Centralized Control in Google’s Datacenter Network”, *ACM SIGCOMM Computer Communication Review (CCR)*, vol. 45, no. 4, pp. 183–197, 2015.

[47] I. F. Akyildiz, A. Lee, P. Wang, M. Luo, and W. Chou, “A Roadmap for Traffic engineering in SDN-OpenFlow Networks”, *Elsevier Computer Networks (ComNet)*, vol. 71, pp. 1–30, 2014.

[48] L. E. Doyle, P. D. Sutton, K. E. Nolan, *et al.*, “Experiences from the Iris Testbed in Dynamic Spectrum Access And Cognitive Radio Experimentation”, in *IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2010, pp. 1–8.

[49] J. Mitola, “Cognitive Radio for Flexible Mobile Multimedia Communications”, in *IEEE International Workshop on Mobile Multimedia Communications (MoMuC)*, 1999, pp. 3–10.

[50] R. Abhishek, D. Tipper, and D. Medhi, “Network Virtualization and Survivability of 5G Networks: Framework, Optimization Model, and Performance”, in *IEEE GLOBECOM Workshops (GC Wkshps)*, 2018, pp. 1–6.

[51] A. Blenk, A. Basta, M. Reisslein, and W. Kellerer, “Survey on Network Virtualization Hypervisors for Software Defined Networking”, *IEEE Communications Surveys & Tutorials (COMST)*, vol. 18, no. 1, pp. 655–685, 2015.

[52] R. Sakamoto, T. Cao, M. Kondo, *et al.*, “Production Hardware Overprovisioning: Real-world Performance Optimization Using an Extensible Power-aware Resource Management Framework”, in *IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2017, pp. 957–966.

[53] S. Ghorbani and B. Godfrey, “Towards Correct Network Virtualization”, *ACM SIGCOMM Computer Communication Review (CCR)*, vol. 44, no. 4, 2014.

[54] C. Liang and F. R. Yu, “Wireless Network Virtualization: A Survey, Some Research Issues And Challenges”, *IEEE Communications Surveys & Tutorials (COMST)*, vol. 17, no. 1, pp. 358–380, 2015.

[55] N. M. K. Chowdhury and R. Boutaba, “Network Virtualization: State of the Art and Research Challenges”, *IEEE Communications magazine (ComMag)*, vol. 47, no. 7, pp. 20–26, 2009.

[56] J. Carapinha and J. Jiménez, “Network Virtualization: A View from the Bottom”, in *ACM Workshop on Virtualized Infrastructure Systems and Architectures (VISA)*, 2009, pp. 73–80.

[57] P. Nayak and R. Ricci, “Detailed Study on Linux Logical Volume Manager”, *Flux Research Group University of Utah*, 2013.

[58] N. F. S. de Sousa, D. A. L. Perez, R. V. Rosa, M. A. Santos, and C. E. Rothenberg, “Network Service Orchestration: A Survey”, *Elsevier Computer Communications (ComCom)*, vol. 142, pp. 69–94, 2019.

[59] C. Rotsos, D. King, A. Farshad, *et al.*, “Network Service Orchestration Standardization: A Technology Survey”, *Elsevier Computer Standards & Interfaces (CSI)*, vol. 54, pp. 203–215, 2017.

[60] R. Mijumbi, J. Serrat, J.-L. Gorricho, *et al.*, “Management and Orchestration Challenges in Network Functions Virtualization”, *IEEE Communications Magazine (ComMag)*, vol. 54, no. 1, pp. 98–105, 2016.

[61] R. Vilalta, A. Mayoral, R. Casellas, R. Martínez, and R. Muñoz, “Experimental Demonstration of Distributed Multi-tenant Cloud/Fog and Heterogeneous SDN/NFV Orchestration for 5G Services”, in *European Conference on Networks and Communications (EuCNC)*, 2016, pp. 52–56.

[62] S. Costanzo, I. Fajjari, N. Aitsaadi, and R. Langar, “A Network Slicing Prototype for a Flexible Cloud Radio Access Network”, in *IEEE Annual Consumer Communications & Networking Conference (CCNC)*, 2018, pp. 1–4.

[63] A. Mayoral, R. Vilalta, R. Casellas, R. Martinez, and R. Munoz, “Multi-tenant 5G Network Slicing Architecture with Dynamic Deployment of Virtualized Tenant Management and Orchestration (MANO) Instances”, in *European Conference on Optical Communication (ECOC)*, 2016, pp. 1–3.

[64] A. Devlic, A. Hamidian, D. Liang, *et al.*, "NESMO: Network Slicing Management and Orchestration Framework", in *IEEE International Conference on Communications Workshops (ICC Workshops)*, 2017, pp. 1202–1208.

[65] A. Kliks, P. Kryszkiewicz, L. Kulacz, *et al.*, "Spectrum Management Application for Virtualized Wireless Vehicular Networks: A Step Toward Programmable Spectrum Management in Future Wireless Networks", *IEEE Vehicular Technology Magazine (VTM)*, vol. 13, no. 4, pp. 94–105, 2018.

[66] M. Richart, J. Baliosian, J. Serrat, and J.-L. Gorricho, "Resource Slicing in Virtual Wireless Networks: A Survey", *IEEE Transactions on Network and Service Management (TNSM)*, vol. 13, no. 3, pp. 462–476, 2016.

[67] L. Xia, S. Kumar, X. Yang, *et al.*, "Virtual WiFi: Bring Virtualization from Wired to Wireless", in *ACM SIGPLAN Notices*, vol. 46, 2011, pp. 181–192.

[68] E. Cuervo, P. Gilbert, B. Wu, and L. P. Cox, "CrowdLab: An Architecture for Volunteer Mobile Testbeds", in *IEEE Internaltional Conference on Communication Systems and Networks (COMSNETS)*, 2011, pp. 1–10.

[69] K. Nakauchi, K. Ishizu, H. Murakami, *et al.*, "Virtual Cognitive Base Station: Enhancing Software-based Virtual Router Architecture With Cognitive Radio", in *IEEE International Conference on Communications (ICC)*, 2012, pp. 2827–2832.

[70] S. S. Kumar, R. Knopp, N. Nikaein, *et al.*, "FLEXCRAN: Cloud Radio Access Network Prototype Using OpenAirInterface", in *International Conference on Communication Systems and Networks (COMSNETS)*, 2017, pp. 421–422.

[71] R. Kokku, R. Mahindra, H. Zhang, and S. Rangarajan, "NVS: A Substrate for Virtualizing Wireless Resources in Cellular Networks", *IEEE/ACM Transactions on Networking (TNET)*, vol. 20, no. 5, pp. 1333–1346, 2012.

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

[73] K. Tan, H. Shen, J. Zhang, and Y. Zhang, "Enable Flexible Spectrum Access with Spectrum Virtualization", in *IEEE Dynamic Spectrum Access Networks (DYSPAN)*, 2012, pp. 47–58.

[74] J. Mendes, X. Jiao, A. Garcia-Saavedra, F. Huici, and I. Moerman, "Cellular Access Multi-tenancy Through Small-cell Virtualization and Common RF Front-end Sharing", *Elsevier Computer Communications (ComCom)*, vol. 133, pp. 59–66, 2019.

[75] A. Rostami, P. Ohlen, K. Wang, *et al.*, "Orchestration of RAN and Transport Networks for 5G: An SDN approach", *IEEE Communications Magazine (ComMag)*, vol. 55, no. 4, pp. 64–70, 2017.

[76] The Linux Foundation. (Nov. 2017). Open Network Automation Platform, [Online]. Available: `https://www.onap.org/`.

[77] The Linux Foundation. (2000). Open Source Ecosystems, [Online]. Available: `https://www.linuxfoundation.org/`.

[78] European Telecommunications Standards Institute, "ETSI GR NFV 003: Network Functions Virtualisation (NFV); Terminology for Main Concepts in NFV", ETSI, Tech. Rep., version 1.5.1, Jan. 2020.

[79] A. Maeder, M. Lalam, A. De Domenico, *et al.*, "Towards a Flexible Functional Split for Cloud-RAN Networks", in *European Conference on Networks and Communications (EuCNC)*, 2014, pp. 1–5.

[80] J. F. Santos, M. Kist, J. van de Belt, J. Rochol, and L. DaSilva, "Towards Enabling RAN as a Service - The Extensible Virtualisation Layer", in *IEEE International Conference on Communications (ICC)*, May 2019.

[81] J. F. Santos, M. Kist, J. Rochol, and L. A. Da Silva, "Virtual Radios, Real Services: Enabling RANaaS Through Radio Virtualisation", *IEEE Transactions on Network and Service Management (TNSM)*, vol. 17, no. 4, pp. 2610–2619, 2020.

[82] H. Wen, P. K. Tiwary, and T. Le-Ngoc, "Current trends and Perspectives in Wireless Virtualization", in *IEEE International Conference on Selected Topics in Mobile and Wireless Networking (MoWNeT)*, 2013, pp. 62–67.

[83] F. Fu and U. C. Kozat, "Wireless Network Virtualization as a Sequential Auction Game", in *IEEE International Conference on Computer Communications (INFOCOM)*, 2010, pp. 1–9.

[84] E. Blossom, "GNU Radio: Tools for Exploring the Radio Frequency Spectrum", *The Linux Journal*, vol. 2004, no. 122, p. 4, 2004.

[85] P. Hintjens, *ZeroMQ: Messaging for Many Applications*. O'Reilly, 2013.

[86] D. Lau, J. Blackburn, and J. A. Seely, “The Use of Hardware Acceleration in SDR Waveforms”, in *WInnF Software Defined Radio Technical Conference (SDR)*, 2005, pp. 13–17.

[87] K. Tan, Y. He, H. Shen, J. Zhang, and Y. Zhang, “Spectrum Virtualization Layer”, *Microsoft Research, Tech. Rep. MSR-TR-2011-108*, 2011.

[88] J. F. Santos, D. d. S. Brilhante, J. F. d. Rezende, N. Marchetti, and L. A. DaSilva, “Optimal Embedding of Heterogeneous RAN Slices for Technology-agnostic RANaaS”, *submitted to IEEE Wireless Communications Letters (WCL)*, 2021.

[89] X. Zhang, H. Qian, K. Zhu, R. Wang, and Y. Zhang, “Virtualization of 5G Cellular Networks: A Combinatorial Double Auction Approach”, in *IEEE Global Communications Conference (GLOBECOM)*, 2017, pp. 1–6.

[90] M. Yang, Y. Li, L. Zeng, D. Jin, and L. Su, “Karnaugh-map Like Online Embedding Algorithm of Wireless Virtualization”, in *IEEE International Symposium on Wireless Personal Multimedia Communications (WPMC)*, 2012, pp. 594–598.

[91] J. Van De Belt, H. Ahmadi, and L. E. Doyle, “A Dynamic Embedding Algorithm for Wireless Network Virtualization”, in *IEEE Vehicular Technology Conference (VTC)*, 2014, pp. 1–6.

[92] A. Aijaz, “Hap-SliceR: A Radio Resource Slicing Framework for 5G Networks With Haptic Communications”, *IEEE Systems Journal (SJ)*, vol. 12, no. 3, pp. 2285–2296, 2017.

[93] G. Sun, Z. T. Gebrekidan, G. O. Boateng, D. Ayepah-Mensah, and W. Jiang, “Dynamic Reservation and Deep Reinforcement Learning Based Autonomous Resource Slicing for Virtualized Radio Access Networks”, *IEEE Access*, vol. 7, pp. 45 758–45 772, 2019.

[94] C. Sexton, N. Marchetti, and L. A. DaSilva, “Customization and Trade-offs in 5G RAN Slicing”, *IEEE Communications Magazine (ComMag)*, vol. 57, no. 4, pp. 116–122, 2019.

[95] A. F. Demir and H. Arslan, “Inter-numerology Interference Management with Adaptive Guards: A Cross-layer Approach”, *IEEE Access*, vol. 8, pp. 30 378–30 386, 2020.

[96] S. Eldessoki, D. Wieruch, and B. Holfeld, “Impact of Waveforms on Coexistence of Mixed Numerologies in 5G URLLC Networks”, in *VDE International ITG Workshop on Smart Antennas (WSA)*, 2017, pp. 1–6.

[97] A. Lodi, S. Martello, and M. Monaci, “Two-dimensional packing problems: A survey”, *Elsevier European Journal of Operational Research (EJOR)*, vol. 141, no. 2, pp. 241–252, 2002.

[98] J. Egeblad and D. Pisinger, “Heuristic Approaches for the Two-and Three-dimensional Knapsack Packing Problem”, *Elsevier Computers & Operations Research (COR)*, vol. 36, no. 4, pp. 1026–1049, 2009.

[99] C.-Y. Chang, N. Nikaein, and T. Spyropoulos, “Radio Access Network Resource Slicing for Flexible Service Execution”, in *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2018, pp. 668–673.

[100] R. Katona, V. Cionca, D. O’Shea, and D. Pesch, “Virtual Network Embedding for Wireless Sensor Networks Time-Efficient QoS/QoI-Aware Approachi”, *IEEE Internet of Things Journal (IoT)*, vol. 8, no. 2, pp. 916–926, 2020.

[101] P. Lv, X. Wang, and M. Xu, “Virtual Access Network Embedding in Wireless Mesh Networks”, *Elsevier Ad Hoc Networks (Ad Hoc Netw)*, vol. 10, no. 7, pp. 1362–1378, 2012.

[102] G. Bhanage, Y. Zhang, and D. Raychaudhuri, “Virtual Wireless Network Mapping: An Approach to Housing MVNOs on Wireless Meshes”, in *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2011, pp. 187–191.

[103] P. Weitkemper, J. Bazzi, K. Kusume, A. Benjebbour, and Y. Kishiyama, “Adaptive Filtered OFDM with Regular Resource Grid”, in *IEEE International Conference on Communications Workshops (ICC)*, 2016, pp. 462–467.

[104] G. Pataki, “Teaching Integer Programming Formulations Using the Traveling Salesman Problem”, *SIAM Review (SIREV)*, vol. 45, no. 1, pp. 116–123, 2003.

[105] M. E. Kurz, “Heuristics for the Traveling Salesman Problem”, *Wiley Encyclopedia of Operations Research and Management Science*, 2010.

[106] M. B. Abdulrazaq, Y. S. Tahir, S. Sha’aban, and M. S. Jibia, “Polynomial Reduction of TSP to Freely Open-loop TSP”, in *IEEE International Conference of the IEEE Nigeria Computer Chapter (NigeriaComputConf)*, 2019, pp. 1–4.

[107] H. H. Chieng and N. Wahid, “A Performance Comparison of Genetic Algorithm’s Mutation Operators in n-cities Open Loop Travelling Salesman Problem”, in *Springer Recent Advances on Soft Computing and Data Mining*, 2014, pp. 89–97.

[108] C. Nilsson, “Heuristics for the Traveling Salesman Problem”, *Linkoping University*, vol. 38, pp. 00 085–9, 2003.

[109] B.-I. Kim, J.-I. Shim, and M. Zhang, “Comparison of TSP Algorithms”, *Project for Models in Facilities Planning and Materials Handling*, vol. 1, pp. 289–293, 1998.

[110] B. Golden, L. Bodin, T Doyle, and W Stewart Jr, "Approximate Traveling Salesman Algorithms", *INFORMS Operations Research (OpRe)*, vol. 28, no. 3-part-ii, pp. 694–711, 1980.

[111] Gurobi Optimization, Inc, *Gurobi Optimizer Reference Manual*, 2021. [Online]. Available: `http://www.gurobi.com`.

[112] J. F. Santos, J. van de Belt, W. Liu, *et al.* (Sep. 2018). Orchestrating Next-Generation Services Through End-to-End Network Slicing, The ORCA Consortium, [Online]. Available: `https://orca-project.eu/wp-content/uploads/sites/4/2018/10/orchestrating_e2e_network_slices_Final.pdf`.

[113] J. F. Santos, W. Liu, X. Jiao, *et al.*, "Breaking Down Network Slicing: Hierarchical Orchestration of End-to-End Networks", *IEEE Communications Magazine (ComMag)*, vol. 58, no. 10, pp. 16–22, 2020.

[114] N. Nikaein, E. Schiller, R. Favraud, *et al.*, "Network Store: Exploring Slicing in Future 5G Networks", in *ACM International Workshop on Mobility in the Evolving Internet Architecture (MobiArch)*, 2015, pp. 8–13.

[115] A. Fox, S. D. Gribble, Y. Chawathe, E. A. Brewer, and P. Gauthier, "Cluster-based Scalable Network Services", in *ACM Symposium on Operating Systems Principles (SOSP)*, 1997, pp. 78–91.

[116] R. Rosen, "Linux Containers and the Future Cloud", *The Linux Journal*, vol. 240, no. 4, pp. 86–95, 2014.

[117] J. Xianjun, L. Wei, and M. Michael. (Sep. 2020). Open-source IEEE802.11/Wi-Fi Baseband chip/FPGA Design, [Online]. Available: `https://github.com/open-sdr/openwifi`.

[118] B. Pfaff, J. Pettit, T. Koponen, *et al.*, "The Design and Implementation of Open vSwitch", in *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2015, pp. 117–130.

[119] F. Tomonori, "Introduction to Ryu SDN Framework", *Open Networking Summit*, pp. 1–14, 2013.

[120] (Feb. 2021). PyLXD – A Python library for interacting with the LXD REST API, Linux Containers, [Online]. Available: `https://github.com/lxc/pylxd`.

[121] K. Zheng, Z. Yang, K. Zhang, *et al.*, “Big Data-Driven Optimization for Mobile Networks Toward 5G”, *IEEE Network*, vol. 30, no. 1, pp. 44–51, 2016.

[122] Y. Shi, J. Zhang, K. B. Letaief, B. Bai, and W. Chen, “Large-Scale Convex Optimization for Ultra-Dense Cloud-RAN”, *IEEE Wireless Communications (WCM)*, vol. 22, no. 3, pp. 84–91, 2015.

[123] J. Roh, Y. Ji, Y. G. Lee, and T. Ahn, “Femtocell Traffic Offload Scheme for Core Networks”, in *IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, 2011, pp. 1–5.

[124] A. Gorod, R. Gove, B. Sauser, and J. Boardman, “System of Systems Management: A Network Management Aproach”, in *IEEE International Conference on System of Systems Engineering (SoSE)*, 2007, pp. 1–5.

[125] R. Ghosh and K. Lerman, “Structure of Heterogeneous Networks”, in *SIAM International Conference on Computational Science and Engineering (CSE)*, vol. 4, 2009, pp. 98–105.