Department of Communications and Networking

Software-defined Communication Platform Implementation on Commodity Hardware

Nicolas Malm

Aalto University

DOCTORAL THESES

Aalto University publication series
DOCTORAL THESES 95/2022

Software-defined Communication Platform Implementation on Commodity Hardware

Nicolas Malm

A doctoral thesis completed for the degree of Doctor of Science (Technology) to be defended, with the permission of the Aalto University School of Electrical Engineering, at a public examination held at the lecture hall TU1 of the school on 12 August 2022 at 12:00.

Aalto University
School of Electrical Engineering
Department of Communications and Networking

Supervising professor
Professor Olav Tirkkonen, Aalto University, Finland

Thesis advisor
Doctor Kalle Ruttik, Aalto University, Finland

Preliminary examiners
Professor Florian Kaltenberger, EURECOM, France
Doctor Anatolij Zubow, Technische Universität Berlin, Germany

Opponent
Professor Leonardo Cardoso, Institut national des sciences appliquées de Lyon, France

Aalto University publication series
DOCTORAL THESES 95/2022

ISBN 978-952-64-0867-5 (printed)
ISBN 978-952-64-0868-2 (pdf)
ISSN 1799-4934 (printed)
ISSN 1799-4942 (pdf)
http://urn.fi/URN:ISBN:978-952-64-0868-2

Unigrafia Oy
Helsinki 2022

Finland

NORDIC SWAN ECOLABEL
Printed matter
4041-0619

Aalto University

Abstract

Aalto University, P.O. Box 11000, FI-00076 Aalto www.aalto.fi

Author
Nicolas Malm

Name of the doctoral thesis
Software-defined Communication Platform Implementation on Commodity Hardware

Publisher School of Electrical Engineering

Unit Department of Communications and Networking

Series Aalto University publication series DOCTORAL THESES 95/2022

Field of research Telecommunications

Manuscript submitted 23 June 2021 **Date of the defence** 12 August 2022

Permission for public defence granted (date) 5 July 2022 **Language** English

☐ **Monograph** ☒ **Article thesis** ☐ **Essay thesis**

Abstract

Radio access networks face increasingly diversified and strict demands from applications. The diverse needs of users cannot be satisfied by a single approach. Networks must evolve into flexible platforms offering per-user service customized to the needs of each application. Meeting the increased coverage and capacity demands of applications also requires network densification. Mobile network operators are confronted with the need to improve their service offerings while keeping the capital and operational expenditure of densified networks under control. Expanding the network using traditional base station designs presents cost and interference challenges. Introducing a functional split between central and distributed units enables better co-ordination and cheaper distributed units situated closer to cell sites.

This thesis contributes to software-based soft—real-time radio access networks (RAN) implementation using commodity hardware. Techniques are presented for implementing software-defined radio nodes effectively. The overall aim is to exploit the benefits of commodity hardware while mitigating the challenges caused by its focus on throughput and polyvalency over low latency and determinism. This work details an architecture for decoupling the communication protocol code from the underlying platform. The approach used is to hide and recover from deadline misses instead of being overly conservative in an effort to provide guarantees. The benefits afforded by increased flexibility in the RAN outweigh occasional lost data due to deadline overruns. The strict requirements on latency imposed by the time domain structure of air interface protocol creates challenges for power management. A method for latency-aware power management is proposed to help solve this. Building on the above, this thesis also investigates disaggregated multi-node RAN implementations. Dividing RAN functionality into physically separate nodes introduces additional latency. The midhaul linking these nodes together becomes an important factor of performance analysis. The research in this thesis presents models for characterizing the behaviour of the midhaul. Models provide tools to assess which functional splits are suitable for a particular scenario.

The validity of the techniques and models presented in this thesis are verified through the use of testbeds. Prototype implementations show the viability of building soft—real-time software-based RANs on commodity hardware. The feasibility of midhaul-based disaggregated designs is demonstrated through the use of user equipment positioning as a test application. The results obtained show soft—real-time commodity hardware designs can enable novel RAN architectures.

Keywords Software-defined Radio, Cloud Radio Access Networks, Cellular Network Architectures

ISBN (printed) 978-952-64-0867-5 **ISBN (pdf)** 978-952-64-0868-2

ISSN (printed) 1799-4934 **ISSN (pdf)** 1799-4942

Location of publisher Helsinki **Location of printing** Helsinki **Year** 2022

Pages 178 **urn** http://urn.fi/URN:ISBN:978-952-64-0868-2

Preface

Work for this doctoral thesis was carried out at the Department of Communications and Networking (ComNet). It provided an excellent environment to learn and perform research while enjoying the experience. In particular, the many practical projects have been greatly appreciated. Many individuals have supported the completion of this work and deserve thanks.

Professor Olav Tirkkonen for his patience and rigour during my doctoral studies. Exacting writing standards, in particular, have helped me progress significantly. My advisor, Doctor Kalle Ruttik, provided a likewise valuable contribution through plentiful creative ideas. Discussions on software engineering have proved enlightening. The Support Group handled with skill and dedication the many and varied tickets I wrote. Viktor Nässi runs an excellent laboratory. He always went beyond the call of duty to furnish all necessary equipment and setups. I had the pleasure to collaborate with Professor Riku Jäntti and his group on many interesting projects. They also made for competent co-authors who were a pleasure to work with. Jussi Kertula, Liang Zhou and Estifanos Menta all helped to produce the results contained in this dissertation.

Directly or indirectly, Yihenew Beyene, Jari Lietzén, Roni Fagerholm, Veli-Matti Rantanen and Norshahida Saba also contributed not only to the completion of my studies but also to make working at ComNet enjoyable and educational. There was less opportunity to interact with other colleagues but I still remember the interesting discussions fondly.

Others, from outside ComNet, have also contributed to this dissertation. The pre-examiners provided me diligent assessment and valuable feedback. Friends and family helped out each in their own way. Keeping daily life running smoothly ensured I could focus on my work.

Espoo, July 17, 2022,

Nicolas Malm

Contents

List of Publications

This thesis consists of an overview and of the following publications which are referred to in the text by their Roman numerals.

I Jussi Kerttula, Nicolas Malm, Kalle Ruttik, Riku Jäntti and Olav Tirkkonen. Implementing TD-LTE as software defined radio in general purpose processor. In *Proc. ACM Workshop of Software Radio Implementation Forum*, pp. 61–67, August 2014.

II Nicolas Malm, Kalle Ruttik and Olav Tirkkonen. Latency-Aware Power Management in Software-Defined Radios. *Journal of Electrical and Computer Engineering*, Volume 2020 pp. 1–19, February 2020.

III Nicolas Malm, Liang Zhou, Estifanos Menta, Kalle Ruttik, Riku Jäntti, Olav Tirkkonen, Mário Costa and Kari Leppänen. User Localization Enabled Ultra-dense Network Testbed. In *Proc. IEEE 5G World Forum*, pp. 405–409, July 2018.

IV Estifanos Yohannes Menta, Nicolas Malm, Riku Jäntti, Kalle Ruttik, Mário Costa and Kari Leppänen. On the Performance of AoA based Localization in 5G Ultra Dense Networks. *IEEE Access*, Volume 7 pp. 33870–33880, March 2019.

V Nicolas Malm, Kalle Ruttik and Olav Tirkkonen. Midhaul Performance Modelling Using Commodity Hardware C-RAN Testbed. Submitted to *Wireless Personal Communications*, February 2022.

VI Nicolas Malm and Olav Tirkkonen. Scheduling Latency of Midhaul-based Commodity Hardware C-RAN. *IEEE 4th 5G World Forum (5GWF)*, pp. 7-12, November 2021.

Author's Contribution

Publication I: "Implementing TD-LTE as software defined radio in general purpose processor"

The author contributed to the analysis of the problem domain, design of the presented technique and writing of the paper. The author also had a major role in the implementation of the testbed and is one of the co-inventors in the patent based on the research.

Publication II: "Latency-Aware Power Management in Software-Defined Radios"

The author had the main responsibility for planning the measurements, collecting and analyzing the data, developing the model and writing the paper.

Publication III: "User Localization Enabled Ultra-dense Network Testbed"

The author had the main responsibility for developing the testbed implementation, assessing its non-positioning related performance and writing the paper.

Publication IV: "On the Performance of AoA based Localization in 5G Ultra Dense Networks"

The author contributed to testbed implementation, measurement gathering and generating ideas to handle hardware non-idealities, as well as to

writing the paper.

Publication V: "Midhaul Performance Modelling Using Commodity Hardware C-RAN Testbed"

The author had the main responsibility for planning the measurements, collecting and analyzing the data, developing the model and writing the paper.

Publication VI: "Scheduling Latency of Midhaul-based Commodity Hardware C-RAN"

The author had the main responsibility for planning the measurements, collecting and analyzing the data, developing the model and writing the paper.

Abbreviations

ACK Acknowledgement

AoA Angle of arrival

BBU Baseband unit

BS Base station

CAPEX Capital expenditure

CFN Cell-free network

CoMP Co-ordinated multi-point

CPU Central processing unit

CSI Channel state information

CU Central unit

DU Distributed unit

GPOS General-purpose operating system

GPP General-purpose processor

IoT Internet of things

IT Information technology

LO Local oscillator

LoS Line of sight

LTE Long-term Evolution

MBB Mobile broadband

MAC Medium access control

MEC Mobile edge cloud

MM Mobility management

NACK Negative acknowledgement

NIC Network interface card

OPEX Operational expenditure

OS Operating system

PTP Precision Time Protocol

QoE Quality of experience

QoS Quality of service

RAN Radio access network

RF Radio frequency

RRH Remote radio head

RTOS Real-time operating system

RU Radio unit

RX Reception

SDR Software-defined radio

TTI Transmission time interval

TVD Total variation distance

TX Transmission

UCN User-centric network

UDN Ultra-dense network

UE User equipment

VHEL Virtual hardware enhancement layer

WCET Worst-case execution time

Symbols

$\boldsymbol{c}$ System-specific coefficients

C_{i} System-specific coefficient

d_{tv} Total variation distance

f CPU frequency

f_{mean} Mean DU reporting latency

f_{std} Standard deviation of DU reporting latency

F_{P} Total processing time

N_{DU} Number of DUs

P_{out} Target late TTI rate

R_{l} Late TTI rate

R_{s} Sampling rate

S_{i} Markov chain latency class

T_{BS} BBU processing time

T_{FH} Fronthaul transmission latency

T_{OS} OS delays

μ Log-normal distribution parameter

σ Log-normal distribution parameter

1. Introduction

Since its inception, use of cellular technology has continuously expanded into new areas. This trend is expected to continue with ongoing fifth generation (5G) deployment [3][78] and into the sixth generation (6G) [101][38]. Each generation expands the capabilities offered while targeting key use cases. First generation (1G) and second generation (2G) cellular systems aimed to provide wireless voice communication between humans, later adding messaging and rudimentary data capability [28][43]. 2G essentially operated as an extension of landlines. The third generation (3G) focused on data connectivity as a primary use case. Support for basic multimedia services were added to cater to still-limited terminals. The fourth generation (4G) greatly enhanced performance and reduced latency while modernising the structure of the mobile core network. Furthermore, the architecture become fully Internet Protocol based to better accommodate enhanced terminal capabilities and emerging applications. Alongside the main mobile broadband (MBB) use case, support was added for terminals with low computational power and energy consumption. Along with these evolutions, cellular network have spread to provide connectivity over almost all inhabited areas. The ubiquity of networks and smartphones has led to many essential, daily services being operated via cellular networks.

Development of 5G [3] explicitly aimed at expanding the range of applications supported. In particular, it aims at moving beyond human-centric and best-effort services. Many automation and industrial applications employ vastly different communication patterns than legacy applications while requiring greater reliability. Another target was a significant reduction in latency to 0.5 ms for the data plane. As cellular networks become a pervasive utility in the future, a single design can no longer fulfil all needs. Systems must therefore diversify to provide increased performance within the constraints of the scarce electromagnetic spectrum. Higher frequency bands will offer larger available bandwidths but more challenging propagation properties. Additionally, to provide sufficient capacity and homogeneous coverage, networks are densifying. The resultant networks will need to manage greater numbers of nodes and frequency bands

simultaneously.

Hosting applications directly on cellular networks reshapes the relationship between stakeholders [113][69]. Traditional networks have been deployed by operators to provide a set offering to customers. Future networks are expected to enable subdivision into virtual instances, each individually configurable by its user. Applications will be able to exploit communication, computation, positioning and other services on a single platform. Sharing information on a single platform provides more efficiency than using disparate systems. Integrated machine learning solutions provide an opportunity to perform network planning and configuration more effectively to fully exploit the capabilities of infrastructure.

Handling the increasing complexity and sophistication of networks, while widening the service offering, benefits from the increased use of software in communication system implementation. Software-based platforms differ markedly from traditional hardware-centric designs [41][50]. Notably, they reduce costs by enabling gradual development through updates post-deployment instead of requiring meticulous verification of all intended functionality during the design phase. Managing complexity and heterogeneity will take an added importance as cellular systems evolve beyond mere access networks to offer additional services to an increasing degree, such as computational edge clouds and positioning.

Softwarization acts as an enabler to increase the flexibility and configurability of radio access networks (RAN) [94]. A software-based approach enables RANs to be assembled from readily available commodity equipment and tailored via configuration changes only. The OpenAirInterface (OAI) project has demonstrated evolving from 4G to 5G and from small installation to large scale distributed deployments [55][56]. OAI and other software-based testbeds have been adopted for research to conduct experimental studies that would have been very challenging to conduct with traditional fixed-functionality equipment. Interest in enabling the benefits of softwarization also exists in the industry, as evidenced by groups such as the O-RAN Alliance [9] and Open RAN Policy Coalition [24]. The concept of open RAN aims to improve cellular networks through flexibility, innovation and efficiency via increased interoperability. Diverse equipment is envisioned to be controlled via standardized and open software interfaces. Tailoring service offerings to the need of each potential application can help address more markets and use-cases.

1.1 Motivation and Objectives

Softwarization enables many benefits contingent on the challenges being overcome. Identifying the challenges of softwarization requires study of the properties of software-defined radio (SDR) platforms and the loads

intended to run on them. Characterization of platforms helps design suitable architectures. Architecture design involves selecting components and how to interconnect them to successfully implement a softwarized cellular network. In addition to the platform, the behaviour of cellular network implementations as a processing load must also be understood. Cloud–radio-access-network (C-RAN) performance analysis serves a similar purpose as channel models. A proper understanding of the characteristics C-RAN processing allows for creating solutions that respond effectively to those needs. The behaviour of platforms and loads will jointly determine appropriate software engineering approaches. The aim is to fully exploit strengths of the platform while mitigating its weaknesses. Processing tasks need to be assigned to the platform component most suitable for them.

Furthermore, as cellular networks become increasingly a component of all systems, the interaction of each autonomously configured component will influence the overall performance of the whole system. Stakeholders require performance guarantees regardless of other users' activity. Vertical applications and multiple services providers can operate on a single C-RAN platform. Their co-existence must be understood to effectively realise infrastructure sharing. As programmable infrastructure platforms, cellular networks will need to provide arbitration and quality of service (QoS) guarantees in addition to flexibility. The envisioned benefits of networks can only be reached if implementations can meet its targets. In particular, the use of general-purpose processor (GPP) clouds and commodity server hardware provides an attractive approach in terms of cost. A network built upon fully interchangeable infrastructure nodes would enable allocating resources to software-defined networking, network function virtualization, C-RAN and user applications with maximal flexibility in response to changing demands. Studies into the performance of soft–real-time SDR on commodity hardware are therefore needed. Reaping the benefits C-RAN and future network architecture requires implementations to be suitable for real-world usage. Instead of selecting a design first and assuming that the implementation will follow, the characteristics of the technology used must be understood in order to be fully utilized.

1.2 Research Methodology

Research conducted for this thesis followed a testbed-based experimental methodology. Testbeds provide validation of design assumptions against reality. Moreover, experiments provide insight into important design considerations. C-RANs are complex systems with various factors such as hardware characteristics, power consumption behaviour, latency limitations, application requirements and resource constraints. This complexity

makes closed form modelling and optimization intractable in practice. The myriad combinations of software and hardware make modelling and predictions difficult.

The vast number of hardware-software combinations and configurations in existence prohibit exhaustive testing. Models were therefore generated to generalize the results obtained. In particular, methods to build adaptive and learning models were studied in an effort to accommodate a wider range of equipment than those tested directly. The testbeds developed used different system configurations to enable comparative analysis and illustrate the ability of the presented methods to adapt. Moreover, the viability of the testbed design was validated in measurement campaigns.

1.3 Structure and Contributions

This thesis contributes to the study of software-defined cellular radio platforms. Contributions are made to the characterization of C-RAN performance on commodity hardware. Factors affecting the performance of SDR platforms are identified based on the experiments conducted. Furthermore, design techniques to mitigate issues and exploit the potential advantages are presented. Observations on practical challenges offer guidelines for future implementations.

The cornerstone contribution of this work is the virtual hardware enhancement layer (VHEL). The VHEL hides the non-idealities of the platform from the communication protocol implementation. Doing so decouples communication implementations from the hardware and operating system (OS) they run on. This, in turn, simplifies implementations while enabling porting to different underlying execution platforms, physical or virtual. VHEL-based implementations are thus suitable for use in centralized baseband unit (BBU) pools as called for in the C-RAN concept.

Contributions are made to the measurement and modelling of C-RAN performance. The metric of late transmission time intervals (TTI) is used to assess the performance of implementations. Due to the vast number of possible hardware, software and configuration combinations possible, comprehensive platform characterization studies are infeasible. To solve this problem, this thesis presents a generic framework tunable for each particular setup. A study on the trade-off between deadline miss rate and energy concepts makes use of the model. Power management is studied from the viewpoint of the constraints imposed by the nature of time-division duplex communication systems. Setting appropriate operating points for C-RAN platforms can reduce the operational expenditure (OPEX) of infrastructure providers significantly [68] while maintaining acceptable QoS.

Expanding on the results obtained for the implementation of SDR sys-

tems on single nodes, this thesis also contributes to the study of disaggregated RAN architectures. Disaggregated architectures split functionality among multiple physical nodes. These nodes exchange information to jointly perform the tasks requried of a RAN. A central unit (CU) provides centralized maanagement and decision making. A CU managing multiple distributed units (DU) can make more efficient use of available resources, notably spectrum, through its view of the overall state of the network. In such distributed implementations, the ability of a single node to meet its execution deadlines does not, by itself, suffice to ensure adequate performance. Even correct information delivered too late can yield poor decisions. Thus, the ability of a disaggregated architecture to effectively synchronize and share information constitutes an important metric.

This thesis focuses on designs where the RAN stack decision-making making is split between CU and DU. In particular, Option 5 from TR38.801 [1], which the divides the Medium Access Control (MAC) layer into two parts communicating over the midhaul. The midhaul links the nodes of the RAN together and plays a critical role in overall performance in the studied functional split. Understanding its latency behaviour enables designing appropriate algorithms and network structures to exploit distributed platforms. Midhaul performance impact is assessed using user positioning as a test use case. Results presented on positioning performance highlight both the potential and challenges of distributed implementations. Location error models provide a tool for scaling network infrastructure resources to obtain a desired level of performance.

This thesis is organized as follows. Chapter 2 provides an overview of RANs and their development to their present form. The main tasks and functions of the RAN for enabling wireless communication will be presented. This functionality constitutes the workload that must successfully transferred unto C-RAN platforms. Furthermore, C-RAN provides inherent advantages beyond a more cost-effective and flexible implementation of RANs. A C-RAN approach enables greater efficiency through joint optimization between RAN functionality and services running on the network. Chapter 3 presents contributions of this work to the study and implementation of SDRs on commodity hardware. Measurements and observations based on testbeds are reported. In addition, architectures and design techniques to address the constraints are presented. Chapter 4 details results on the implementation of disaggregated RAN architectures. Potential future RAN designs serve as the framework for system-level evaluations. Testbed-derived results provide data to build models to quantify the impact of low-level C-RAN platform metrics on overall system performance. Finally, Chapter 5 draws conclusions based on the findings of this work. Avenues for future research are also presented.

1.4 Summary of the Publications

This work comprises an introductory part and six original publications. Publications I–III focus on the implementation of SDRs on commodity hardware at the level of a single node. The performance and characteristics of SDRs as a computational load are discussed and analysed. Publication IV and Publication VI widen the scope to study the behaviour of a collection of nodes working together to implement an SDR-based RAN. The primary service considered in performance evaluation is user equipment (UE) positioning in future RAN architectures.

Publication I, along with patent [54], presents the VHEL concept. The core concept lies in masking the imperfections due to missed deadlines from the protocol code in such a way that performance issues appear as channel fading events. The VHEL hides the impact of late TTIs and corrects local oscillator (LO) offsets. Additionally, a mechanism to transform flow-based in-phase–quadrature (I/Q) samples into TTIs for communication protocol upper-layer processing is presented. A partial implementation of a time-division Long-term Evolution (LTE) BBU serves as the workload to test the VHEL-based platform. A non–real-time OS executing on a GPP provides sufficient performance. The primary metric for performance evaluation was the number of late TTIs encountered per second. This indicates how often the software BBU was late in providing the requisite samples to the frontend. A secondary performance metric was the number of loss-of-synchronization events on the UE side. Loss of synchronization occurs when the BBU fails to send enough TTIs on time to enable the UE to remain synchronized. The publication also reports observations on the late clustering behaviour and influence of LO offset from the viewpoint of LTE protocol processing.

In Publication II, power management within the constraints of the deadlines imposed by the communication protocol is studied. Measurements quantify the relationship between central processing unit (CPU) clock frequency and the number of late TTIs observed. This trade-off leads to a model relating energy consumption to the predicted late rate. The key observation lies in the existence a CPU frequency above which increases in CPU clock speed provide negligible performance gains. Computational complexity of the communicaiton protocol is not the only factor impacting performance. The frequency threshold depends on network processing latency and jitter as the BBU needs to first receive samples from frontends in order to process them. Thus, to further improve energy consumption, hardware-accelerated network processing is employed to reduce the overhead from network interrupt processing. A method to fit the model to other hardware-software configurations using mean squared error and goodness-of-fit is presented.

Publication III applies the single-node architecture of Publication I

and Publication II to a multi-node setup. Processing is split between a CU and two DUs. The communication between CU and DU introduces the need to maintain a common notion of time in order to effectively exchange reports and commands. System state information is distributed between CU and DUs. Different time scales are used in the CU and DUs to accommodate different decision-making periodicities. DUs operate at a time scale defined by the TTI duration of the air interface. Matters handled by the CU update at a slower pace than those of the DUs. Decoupling the update period via a shared logical time enables both the CU and the DUs to operate at a suitable periodicity. Nodes maintain a local clock. They translate from this to the shared logical time when sending messages. Furthermore, a protocol stack design adopting a stateless worker approach is presented. It aims at improving performance scalability as a function of CPU core count. Performance is validated through distributed positioning measurements to validate CU-DU co-ordination performance. To this end, a position-based ultra-dense network (UDN) is used. It employs RAN-side positioning to perform mobility management (MM). It is shown to be a viable approach to alleviate signalling overhead.

In Publication IV, a position-based RAN testbed is used to assess the positioning accuracy of a UDN network. A two-level extended Kalman filter approach splits the computational load between the CU and DUs. Angle-of-arrival (AoA) measurements are made by DUs. These reports are then sent to the CU, which then combines the information into the UE's position. The positioning algorithms function as user applications in a mobile edge cloud (MEC). Computationally slow positioning algorithms are co-located with RAN processing to ease information exchange. The design is able to compensate for non-idealities in commercial off-the-shelf hardware using software. Isolation is maintained between the communication protocol and the positioning application. Slow positioning algorithm computations do not impair communication protocol operation. Observations are made concerning transient error conditions, in particular sample timing error correction. Techniques are presented for the handling of such errors.

Publication V investigates the impact of midhaul latency and jitter on the performance of RANs using the Option 5 functional split. Reporting and command latency measurements are reported. Command latency testing involves end-to-end transmission from CU to UE using radio frontends. DU reporting latency is assessed using an experimental setup using container-based virtualization. DUs each run in a separate container. A midhaul link connects the DUs to a CU. The setup records reporting latency as a function of the number of DUs. In both experiments, a common notion of time among nodes is established using Precision Time Protocol (PTP) [49]. Results therefore include offsets and variation caused by imperfect real-world synchronization. Performance is evaluated using the metric of induced position error due to the latency and jitter in midhaul communication.

The results obtained are extrapolated to larger network sizes by means of simulations using a predictive model generated from the gathered data.

In Publication VI, the impact of midhaul performance on air interface scheduling is studied using empirical data from Publication V. Measured latencies are classified in several classes representing latency ranges. It was observed that TTIs in high latency classes tend to cluster together. Outages are thus longer in duration but fewer in number. A model is fit to the empirical data to estimate the frequency and duration of outages faced by UEs. The model shows that scaling as a function of the number of DUs depends on the target late rate. Results are further analysed in terms of midhaul performance impact on spectral efficiency.

2. Radio Access Networks

2.1 Functions and Tasks

Radio access networks are the part of cellular systems wirelessly linking terminals to the core network. RANs operate within a finite allocation of shared and scarce spectrum. Their design must therefore aim to exploit this limited resource as efficiently as possible to be economically viable. Another important consideration is energy efficiency, especially at the terminal side, as many terminals are battery powered. Consequently, RANs have traditionally [74] been engineered to aggressively put terminals into a partially disconnected energy-conserving sleep state. Use of sleep states comes at the expense of additional wake-up latency and less consistent service quality. The mobility of terminals imposes the need to manage the effect of this mobility. Service must be maintained over geographically large areas, ideally transparently to users.

Functionally, RANs consist of a control plane and a data plane. The control plane carries the signalling traffic between base stations (BS) and UEs while the data plane carries users' payloads. Control plane signalling can be either broadcast or unicast in nature. System-level broadcast information distribution scales independently of the number of UEs connected. UE-specific control messages can constitute a substantial load in cases requiring frequent reconfiguration of parameters; for example, high-speed users changing cells at a rapid rate. Control plane strategy thus varies from generation to generation and depends on anticipated use cases and terminal types. A particular design can effectively support only a limited number of users simultaneously.

BSs manage their connected users separately. While BSs may communicate and collaborate, a UE will receive instructions from only the control plane of its master cell. Co-ordination between neighbouring cells can be beneficial [5], for example, in terms of interference mitigation. The benefits, however, must be weighed against the additional overhead of

communication and the complexity of the resulting optimization problem.

At a high-level, most designs will address the same broad set of functions. BSs will perform radio resource management to allocate resources to UEs. This includes allocating time-frequency blocks via scheduling, assigning identifiers and instructing UEs to vary their transmission power. The overall aim is to maximize the utility obtained from the allocated spectrum. Resource allocations cannot be static as users are mobile. RANs must therefore follow users' changing link conditions and respond accordingly. Fast adaptation, however, requires more signalling. Decisions are therefore made a multiple time granularities. Effectively managing user association and resource allocation forms an important part of ensuring QoS [28][85]. UEs with no data to transmit or receive are placed into an idle state. While in idle state, UEs are not associated with any BS [57]. Changes in cell reselection measurements can be reported at a coarser level, potentially reducing signalling overhead. Resuming communication after an idle period incurs a latency penalty, as the RAN must perform paging, session establishment, security enablement, and QoS configuration. UE type also can play a role. The more information the RAN has concerning a particular device, the more accurately it can predict its behaviour and needs. Low-power Internet-of-things (IoT) devices can tolerate larger delays in network access than UEs operating in latency-sensitive applications, such as automation or traffic management. Different UE types may also exhibit regular transmission patterns. Shifting these cycles to not coincide can help spread load evenly and thus avoid momentary peaks overloading available capacity.

2.2 Architecture

Cell site selection results from the process of network planning [73]. Environmental conditions affect signal propagation and play a large part in determining suitable cell sites to achieve the desired coverage level. Network planning also takes into account the expected capacity needed to satisfy demand. Infrastructure owners balance the need for capacity and coverage against the capital expenditure (CAPEX) and OPEX of deploying additional sites. In the case of mobile network operators, expenditure must be weighed against expected revenues determined by the willingness of users to pay for a particular service [53][85]. RAN networks have also been defined by the technological capabilities and sought to accommodate the primary use cases of their time. In particular, site densities have grown [19] as use of mobile data has grown and expected throughput has increased.

Networks have adopted various types of BSs [48] broadly divided into macro-cells and small cells. Macros cover a larger area and provide service

to mobile users. Commonly, macro cells are subdivided into sectors. The combination of requiring high transmit power and capacity to to serve numerous users results in relatively large installation footprint and cost. Small cells offer additional capacity inside the coverage area of a macro cell. Small cells may operate autonomously or receive instructions from a macro cell. They are also cheaper and smaller to install. Further types of small cell, pico and femto cells, provide indoor coverage. These cells typically support only a small number of users and exist mainly to improve coverage while also providing additional capacity. Cells are grouped in tiers, with the highest tier typically being formed by the macro cells. Lower tiers form from progressively smaller coverage area cells. Capacity-providing lower tiers are deployed only in certain locations as service demand distributes unevenly across geography. Hot-spot areas require substantial capacity and density while other areas require much less.

In 5G, BSs may further be split internally into CUs and DUs [2]. A single CU exists and carries responsibility for the operation of the cell. The CU manages one or more DUs under its control via midhaul links. DUs perform radio frequency (RF) signal transmission and reception. CUs choose which DUs to employ to serve a particular UE. A further split of the functionality of the DU has been studied [87][64][39]. In this scheme, part of the physical layer processing and the RF functionality are moved to an entity separate from the DU called the radio unit (RU). Distribution of RUs provides increased probability of good coverage through the provision of spatial diversity.

Regardless of the cell type, they must be connected to the mobile core network via the backhaul [51]. This connection can be wired or wireless. Wired connections provide greater capacity and reliability but incur greater cost to install. Sufficient backhaul capacity constitutes an essential component of RAN performance. Control plane functions employ the backhaul to perform tasks such as UE authentication and mobility management. On the data plane, traffic routing and prioritization affects the QoS. Gateways in the core network forward data between UEs and external networks.

In addition to nodes mandated by the communication protocol, a centralized management controller may also be present in the core network to oversee the operation of the network. An interface for human network operators is also provided. Controllers collect data and assign operational parameters to each BS to maximize overall network efficiency. Such a configuration mechanism operates at a relatively slow time scale as cells operate largely autonomously in traditional network architectures. Furthermore, multi-tier networks complicate the analysis as their interactions create dependencies. Consequently, tiers cannot be analysed independently.

Reducing CAPEX can be achieved through optimization of the structure of the RAN [73][85]. Serving users with the minimum number of sites

possible saves costs for infrastructure owners. Poor network deployment can result in inter-cell interference, coverage holes and unnecessary energy expenditure. Optimization is complicated by the increasing heterogeneity of networks consisting of components such as: macro and small cells, wired or wireless backhauls, beamforming and different frequency bands in simultaneous use.

In addition to lower deployment costs through reduced CAPEX, C-RANs can potentially aid in easing optimization. C-RANs are designed to centralize information, resources and decision making. They are, therefore, inherently designed to be flexible and adaptable for different use cases. C-RAN structural optimization has been investigated to reduce costs and improve RAN performance [52][11][96][7][30]. Moving processing between BBU pools and remote radio heads (RRH) changes the cost of each node as well as the required dimensioning of the fronthaul links. The more information is centralized, the more fronthaul and midhaul data volumes increase. On the other hand, increased centralization enables more efficient exploitation of resources in the air interface via techniques such as co-ordinated multi-point (CoMP), interference cancellation and intelligent handover decisions [77]. Optimization schemes seek to balance computation, fronthaul cost and communication efficiency. Varying the functional split between BBU pools and RRH provides deployment models suitable for different network topologies and traffic scenarios. For example, a dense urban high-demand scenario with fibre connections available will likely benefit from a different split than a rural low-traffic wireless backhaul scenario. The performance characteristics of C-RAN implementations impact the feasibility of functional splits and co-ordination schemes. BBU pools must have low latency and sufficient computational performance to enable network-level approaches.

2.3 Mobility Management

Wide-area mobility management forms one of the hallmarks of cellular systems [18]. Providing uniform coverage and service quality is challenging due to the complexity of radio channel propagation. Results from measurements have limited temporal utility due to the time dependence of channel conditions in realistic environments. While large scale shadow fading obstacles, such as terrain and buildings, remain static over extended periods of time, fast fading occurs over short time scales. Factors such as vehicles, users and weather change constantly. Consequently, propagation properties depend on the environment considered. Urban, sub-urban and rural networks will experience different conditions and user mobility patterns. Another important factor to consider for mobility management is the carrier frequency used. Propagation characteristics vary as different

frequencies experience varying levels of diffraction, refraction, attenuation and penetration loss.

Previous cellular standard generations have mandated UE measurements of nearby cells. In active state MM, the results of these are then always reported to the serving BS. Mobility management decisions are made by the serving BS based on the reports received. Active UE participation in the mobility process causes interruptions in data connectivity in order to provide measurements gaps. Furthermore, signalling overhead results from the need to send reports to the RAN. Effectively, standards mandate that a particular measured quantity serves as a proxy for cell suitability. Since measurements are subject to random fluctuations, techniques such as averaging, hysteresis and thresholds are employed to attempt to distinguish trends from momentary fluctuations. In idle mode MM, UEs periodically check whether they have moved from one tracking area to another and report to the core network if this is the case. UEs must be partially aware of network topology to know which cells belong to which tracking area. In addition to suitable radio links with target and source cells, a successful handover requires sufficient capacity at the target BS.

Handovers, and their related connectivity interruptions, degrade quality of experience (QoE) and reliability. Thus, a substantial amount of research literature [35][103][110][67][92] attempts to address the issue in various ways. One type of optimization [93][34][47] consists in predicting which cells a particular UE will visit in the future. UEs can then be assigned to the most suitable tier and cell. Prediction can exploit the particular characteristics of different use cases. For example, mobility patterns in high-speed railway scenarios [102][33] are very predictable as they are constrained by the layout of train tracks. Similarly, pedestrian and vehicular users behave differently. Another type of optimization exploits diurnal patterns. During workdays, users head to and from work at certain times of days. This shifts the concentration of demand from area to area in a relatively predictable way. Similar patterns exist at the seasonal level.

In multi-tier networks, UEs can be assigned to cells with differing capacity and coverage characteristics. Devices with high mobility but moderate to low throughput needs could be transferred to macro cells, for instance. Models and data can also be combined with machine learning to equip networks with learning and prediction capabilities. Automated configuration helps manage the very large parameter space as well as to adapt in quasi–real-time to changing conditions. Methods proposed by researchers [102][35][103] not only aim to improve QoE for users but also to reduce signalling load on networks. Changing RAN configuration enables reducing unnecessary measurements and handover attempts.

3. Software-defined Cloud Radio Access Networks

Implementation techniques for RANs evolve over time, driven by technological progress. Functionality described in the previous chapter is implemented using new methods. Software-defined designs aim to replace dedicated hardware with software. Separating software and hardware design provides several advantages. Functionality can be adapted to each use case more easily using only software changes. Further enhancements to implementations are possible even after deployment. Reducing the coupling between hardware and software also enables the use of generic components, providing cost benefits. Software-based commodity hardware platforms present different characteristics than traditional purpose-built platforms. Successful implemantation must account for the specific characteristics of these platforms.

In this chapter, we focus on the implementation of an SDR-based RAN node on commodity hardware, which is one of the two main contributions of this work. Section 3.1 first reviews key concepts and challenges related to SDR node implementation. The design choices of the approach used for this work are then presented in Section 3.2. The remainder of this chapter explains each contribution in detail.

3.1 General-purpose Platforms

GPP platforms emerged in the information technology (IT) sector. Convergence of needs led to a standardization of components and designs. The resulting economies of scale reduce the cost and increase the pace of innovation of commodity equipment as a wider customer base shares the cost of research and development. By their nature, GPP platforms aim to support a wide variety of tasks. Consequently, they provide acceptable performance at most tasks but excel at none. The standardization of hardware interfaces enables a large ecosystem of compatible equipment. Performance can hence be improved through the use of dedicated accelerators such as field-programmable gate arrays or graphics processing units.

Similarly to IT, wireless communication networks can benefit from the availability of low-cost commodity hardware. One of the advantages resides in the ability to consolidate other services onto the same platform in addition to the SDR processing itself, such as operations and maintenance and user applications. Latency can thus be reduced compared to an approach using dedicated devices. Another benefit lies in the greater ease of development [106]. GPP platforms have access to a wide variety of software tools and libraries. Implementation requires less device-specific knowledge. GPP platforms provide greater reusability and upgradability than dedicated hardware. Nodes can be reallocated to new tasks based on demand. Nodes can also be partially upgraded by, for example, adding more or newer accelerators.

Effective use of GPP platforms for cellular network requires solving several key challenges. Firstly, GPP platforms are optimized for throughput in the sense of average performance [86] while C-RANs face stringent latency constraints. Secondly, commodity hardware also lacks analogue input-output interfaces as present in dedicated hardware. Consequently, I/Q samples must be processed elsewhere and moved into the GPP platform.

3.1.1 Soft–Real-Time

Real-time systems must complete processing by set deadlines lest a failure occurs [61][86]. Depending on the consequences of these failures, real-time systems are subdivided into two groups: hard–real-time and soft–real-time. Hard–real-time systems cannot tolerate even a single deadline overrun without risking serious negative consequences. Soft–real-time systems can continue operating in spite of occasional deadline misses, albeit with degraded QoS. They only need to be fast enough on average. Systems are typically over-provisioned to provide a sufficient buffer to reduce the probability of overrun below a desired threshold. The downside to this approach resides in increased cost. Fault-tolerant real-time systems should provide recovery mechanisms for cases where deadline overruns occur [89].

The real-time deadlines of C-RAN systems result from the time-related features of cellular standards. These are mainly the division of time into TTIs and upper limits on processing time before acknowledgements (ACK) or negative acknowledgements (NACK) must be issued. The processing load required per TTI to form an ACK-or-NACK decision depends on the number of transmissions and the quantity of data sent. Greater amounts will require more decoding or encoding. Typically, TTI length remains constant, leading to a periodic structure with varying load. Traditional telecommunication infrastructure implementations have used dedicated hardware and real-time operating systems (RTOS) with hard–real-time constraints. To ensure deadlines are met, challenging and labour-intensive design-time verification must be performed in order to ensure that the

worst-case execution time (WCET) remains below the target [71]. Diversification of use cases, features and device types, combined with increasingly complex and heterogeneous networks, greatly increase the number code-paths to study for WCET and schedulability analysis. The number of possibilities to calculate grows combinatorially with the number of features. Beyond a certain size and complexity, it becomes infeasible to provide hard–real-time guarantees. Moreover, network design evolution towards flexible, user-programmable platforms introduces processing load profiles unknown at design time. Naturally, non–time-critical processing should be separated from the telecommunication protocol implementation to avoid causing deadline misses due to interference. This results in a mixed-criticality system. However, even such a separation does not suffice to ensure fulfilment of timing constraints due to possible interference at the hardware level. Certain user applications also require real-time guarantees of their own. The interaction between applications and the communication stack can become complex if the processing load of the communication protocol stack depends on the application and vice-versa.

3.1.2 General-purpose Operating Systems

Operating systems serve to manage hardware by arbitrating resource use and access among tasks. Different OSs exist to cater to different needs. RTOSs are designed to place determinism above all else [86]. They are intended for hard–real-time tasks required bounded WCET for each function to enable schedulability analysis. General-purpose operating systems (GPOS) are designed to cater to a wide variety of needs. Similarly, to GPP platforms, they are flexible at the cost of excelling at no one workload. This flexibility raises the questions to tunability to suit the particular needs of C-RAN. Enabling successful operation on a combination of GPP platform with GPOS offers the greatest adaptability and access to more cost-efficient commodity hardware to build wireless network infrastructure.

As an alternative to pure RTOSs and GPOSs, dual kernel approaches [4] attempt to combine real-time capability with a best-effort environment within a modified GPOS. A hard–real-time kernel executes tasks with strict deadlines. All other tasks, as well as most system management, is performed by a non–real-time kernel. Communication takes place via interfaces designed to avoid blocking the real-time tasks. Xenomai [104] is a prominent example of a dual kernel approach.

C-RAN processing is an interrupt-heavy workload due to the need to move I/Q samples in and out of the BBU. Latency of data movement forms an critical component of the ability to meet deadlines. RTOS and dual kernel approaches are thus challenging to use for C-RANs. Processing can only begin after the required data has been received from other network

nodes. This is difficult to bound. While it is possible to build the entire network to be hard–real-time, immense effort would be needed to prove the ability of such designs to meet all deadlines even with WCETs. Run-time configurability and adaptability to emerging use cases would be severely limited.

A GPOS can be tuned to the needs of wireless communication platforms. GPOS scheduler selection greatly impacts the latency and jitter behaviour of applications executing on it [90]. Because the state of UEs and other network needs affect the processing load of BBU instances, the scheduler of the communication protocol in the BS could inform the OS scheduler of foreseeable load in similar manner to power management [58]. By selecting and tuning the scheduling configuration of a GPOS, a real-time environment can be imperfectly emulated. In the literature, the SDR application receives a higher scheduling priority than other tasks on the system. This is accomplished by changing the scheduling class to be real-time and by assigning isolated, dedicated CPU cores. The OS scheduler will execute tasks in the real-time class. Only when tasks in the real-time class have all either terminated or blocked, will lower priority classes be considered. Prioritization can also be applied among the threads of the SDR application Thread priority determines which threads obtain CPU time in the case of contention within a task.

The advantage of a GPOS-based design compared to a dual kernel approach comes from the ability to tailor characteristics for each application as opposed to simply choosing either the hard–real-time or best-effort environment. Moreover, there is no difference in the programming tools available; only the task's scheduling configuration is changed.

3.1.3 Virtualization

Virtualization technology is widely employed in the IT sector. Large data centres provide cost-efficient, plentiful computational power with a short lead time. Resources can therefore be applied on-demand. Applying virtualization to telecommunications infrastructure can provide benefits in terms of consolidation, power savings and increased flexibility [82][108]. Virtualization use in C-RAN, however, requires adapting the approach to the particular needs of wireless protocols. Public clouds operated using large data centres exploit economies of scale [100]. Such an approach is unsuitable for C-RAN for two reasons. Firstly, latency between RU and BBU grows too large to meet communication protocol deadlines. This results from the incentive to build few but large centres to exploit economies of scale. Due to the relatively small number of data centres, most cell sites will be located far away from them. Secondly, transferring the data generated by RUs to the centralized cloud would place a momentous burden on midhaul and backhaul links. Multiple levels of cloud platforms can provide

a solution. Latency requirements dictate the placement level of each task in the cloud hierarchy.

Consolidating BBU processing into a BBU pool provides advantages beyond a reduction in operating costs. Co-located BBU instances communicating with each other encounter extremely low latency and jitter as they execute on the same platform. The cost of making mobility management and load balancing decisions is thus greatly reduced. In particular, a software interface between two BBU processes can replace communication over a dedicated link between physical BSs. Doing so enables very low overhead handovers. Similarly, UEs can be moved from one BBU instance to a newly started one to balance load. A single platform can also avoid wasting capacity during off-hours. When demand for communication protocol processing lessens, more resources can be allocated to executing network optimization machine learning tasks. Thus the data collected during high-demand hours will be turned into an improved configuration.

Work into virtualized C-RAN on GPP platforms [63][112] has demonstrated the viability of the concept. Application QoS can serve as a metric for measuring how well the C-RAN system can provide service to each user. This is important to support the division of physical resources in slicing. Each application likely has different requirements. For instance, a video streaming service provides satisfactory service even with interruptions in data delivery if these are imperceptible to the human user. Co-existence of multiple BSs on one platform is essential for achieving centralization gains. Researchers have thus studied the scalability of commodity platforms when running BBUs [42][32].

3.1.4 Fronthaul

A fronthaul links the BBU with frontends. Its function is to transport digitized RF data to and from the frontends. Processing of each TTI requires transportation of samples over the fronthaul. Consequently, networking latency must be taken into account when analysing the ability of a system to meet its deadlines. The communication protocol used on the air interface sets constraints on the maximum allowable response time. Fronthaul delays reduce the time available for processing at the BBU. Hence, fronthaul performance constitutes a critical component of C-RAN platform overall performance.

Fronthaul links can be dedicated or shared. Multiplexing the frontends of multiple DUs over a single link introduces queueing and therefore adds jitter. The same applies for multi-antenna systems where a single DU employs multiple frontends multiplexed over a single link. Dedicated physical links for each frontend could alleviate this problem but engenders very large CAPEX to implement. Traditional frontend protocols, such as the Common Public Radio Interface [26], prove difficult to scale to more

general network topologies due to stringent link requirements and lack of support for different functional splits [81][23][12]. Ethernet provides an attractive alternative due to greater flexibility and integration with existing networks. Use of Ethernet also presents advantages from a cost standpoint but presents challenges in terms of synchronization. Ethernet network interface cards (NIC) are commodity hardware widely available.

Frontends process data as a continuous flow of samples. Ethernet links, on the other hand, operate using discrete frames. Latency analysis must take this factor into consideration; frame size is a system design parameter. Samples arrive in groups with the same latency from the viewpoint of the receiver. Lost frames will also cause groups of samples to be lost. Larger frame sizes present a trade-off. Larger frames engender less overhead as more samples can be processed per interrupt. On the other hand, a larger frame means more delay before BBU processing can start and more missing data if the frame is lost. Frame size also impacts the standard deviation of data arrival time. For most wide bandwidth protocols, a single TTI consists of much more data than can fit into an Ethernet frame. A particular implementation can choose to reduce the delay before samples arrive at the cost of increased numbers of interrupts to process per time unit. Alternatively, fewer frames can be employed to save on processing overhead but suffer from more latency and jitter.

Consolidating BSs onto a single platform compounds the interrupt problem. OS interrupt processing time is independent of the SDR code itself. The delay between frame arrival at the NIC and data availability at the SDR application depends on the OS network stack and the driver of the NIC. Network processing acceleration provide one possible solution to reduce overhead.

3.2 Implementation Design Choices

The overall aim of this work's implementation approach lies in presenting to the communication protocol implementation an idealized operating environment. Abstracting imperfections away provides two key benefits: separation of concerns and ease of portability. Communication protocol code is isolated from the underlying OS and hardware. Processing performance optimization and communication protocol code development can proceed independently.

The best-known cellular system research tool for GPP platforms is OpenAirInterface (OAI) [80][56]. OAI provides protocol implementations of LTE and NR compatible with commercial equipment. Its open-source nature enables researchers to make modfication for the purpose of conducting experiments. Unlike OAI, the platforms used in this thesis do not provide complete implementations of particular standards. Work pre-

sented in this thesis focuses on a particular aspect of the implementation of software-defined wireless communication on commodity platforms. In particular, this work focuses on unmodified mainline kernels using Ethernet-connected frontends. OAI, on the other hand, provides a holistic standards-compliant implementation for experimentation with a complete eNB or gNB. It targets USB and PCIe frontends for real-time operation [31] but also supports the Universal Hardware Driver from Ettus Research used with the widely-used Universal Software Radio Peripheral family of devices, and thus Ethernet connected frontends.

On a technical level, there are a number of differences in the approach taken in this work compared to OAI [44][55]. These can be grouped into three categories: task distribution, information management and the functional split. OAI uses a pipelined designed for PHY processing. In this approach, different parts of the processing flow are handled by different threads. These pass their results onto the next stage in the pipeline. Additionally, channel coding processing is further parallelized over segments. A master thread coordinates the worker threads handling individual segments. In contrast, the approach taken in this work centers around work item queues from which worker threads obtain tasks to perform. A task is either a complete TX or RX TTI to process. A worker thread being late only affects the TTI it has been assigned. On the other hand, dedicating processing of a complete TTI to a single worker removes the possibility of parallel processing. It remains, however, possible to easily use accelerators easily as necessary state for a TTI is contained within one worker thread. Time-consuming, but not time critical, processing will be allocated to background threads. This avoids blocking the time-critical work unnecessarily.

The second difference between OAI and the work in this thesis lies in the manner samples and state information is managed. For RX, OAI pipeline processing begins with a part called Front End Process (FEP). It is responsible for obtaining samples from the frontend. For TX processing, the FEP is the last stage of the procesing pipeline. The approach taken in this work is to separate physical layer processing from the frontend hardware itself using the VHEL. In other words, the VHEL exists as a layer below the physical layer. However, it should be noted that since OAI also dedicates a thread to sending and receiving samples from the frontend, the two approaches are broadly similar.

As a consequence of the differences above, sample data management is also handled differently. OAI employs a global buffer indexed by the frame and subframe number being processed. The method used in the platforms of this thesis is based on a pool of TTI-sized buffers. These are allocated at start-up to prevent costly dynamic memory allocations during run-time. RX worker threads acquire pointers to filled buffers from the VHEL. For TX, the VHEL receives buffers containing the samples to be transmitted and rearranges them in the correct order based on timestamps. After use,

buffers pointers are returned to the pool. If processing becomes delayed, TTI sample data buffer pointers may accumulate in the queues. In this case, older ones are discarded until those that remain have timestamps close enough to the most recently received samples. This avoids performing processing that will be late as the window for the results to be useful has passed. OAI, on the other hand, uses feedback signalling in the pipeline to communicate that a given stage is ready to receive more work.

The third difference in the architecture of OAI and this thesis concerns the functional split [55]. OAI has adopted Option 2 to reflect the choice made by the 3rd Generation Partnership Project (3GPP). The functional split studied in this work is Option 5, which splits the MAC layer. The resulting challenges and architectural choices are discussed in Chapter 4.

The VHEL is introduced in Publication I. It serves as the foundation of the platform abstraction. The VHEL presents a consistent interface to the communication protocol code regardless of the underlying platform used. Deadline misses resulting from the soft–real time nature of the platform will be managed by the VHEL. This further helps to provide a consistent execution environment for the communication protocol code. The cornerstone of the approach in this work is the technology component called VHEL. The basic structure was conceived during the work on Publication I. Further work then used and refined this approach.

As discussed in Section 3.1, networking performance forms an integral part of overall SDR performance. The testbeds of Publications I-VI assign the highest priority to the thread responsible for communicating with frontends. This is done as the incoming sample stream provides the basis for timing and any delays in it would thus jeopardize the functioning of the entire SDR. In Publication II, the impact of network traffic interrupts is considered in terms of C-RAN performance. Offloaded frame processing greatly reduces the jitter in processing times, leading to a more predictable operating environment for the SDR. Publication V uses a software-based kernel bypass technique to achieve latency reduction in a manner similar to hardware offloading without introducing any hardware dependency.

Publication II investigates SDR node adaption to platforms with different performance characteristics. The deadline miss rate is modelled as a function of communication protocol load and CPU clock frequency. Power management can use such a model to select an appropriate CPU frequency to achieve a desired deadline miss rate without wasting energy. CPU clock management is used rather than low-power states due to the timing constraints of the air interface. Certain functions must be executed regularly for the radio link to be maintained. Additionally, not all UE transmissions can be anticipated ahead of time. Since these transmissions require a response within a set time, the RAN must be able to detect them as soon as possible.

Publication III introduces background workers to handle computationally

heavy tasks without strict deadlines. Long-running calculation can be carried out in parallel to TTI processing. Background workers use the computational capacity left over by the higher priority per-TTI workers. When the results are ready, they are transferred to the upper layers of the communication protocol. Timestamping enables the upper layers to assess usability of the results. If the computations completed too slowly, the results can simply be ignored. The system will continue to operate with degraded performance due to the late information but no failure will occur.

Publication V studies performance with container-based virtualization. Multiple DUs are hosted inside Linux Containers. Compared to a full virtual machine, containers are more lightweight , as fewer aspects of the host are virtualized. In particular, the kernel is shared with the host OS. This provides an important advantage for C-RAN platforms as there are fewer layers of networking hardware and network stack virtualization for I/Q samples to traverse.

3.3 Hardware Abstraction

In order to solve the problems presented in Section 3.1, while retaining the advantages of GPP platforms, Publication I presents the VHEL. It is designed to abstract away the non-idealities of the GPP platform from the protocol stack. Communication protocol code does not, therefore, need to handle platform-related non-idealities. The VHEL manipulates I/Q samples as they transit between the frontend and BBUs. Lost incoming samples are replaced by dummy data to appear as channel errors to the protocol code. Late outbound samples are replaced by either zeros or a pre-defined contingency TTI. For example, the contingency TTI could contain only pilots and synchronization signals to help avoid UE loss of synchronization when BBU deadline misses occur. For transmissions, the VHEL will take TTI buffers from the protocol code and send them to the frontends. The VHEL can reorder outgoing samples. This may be necessary when transmission (TX) code is multi-threaded for performance reason. In this case, it is possible that TTI $N+1$ becomes ready ready before TTI N. These will be reordered to ensure that the samples leave in the correct order. Algorithm 1 lists the procedures used for handling TX and reception (RX) TTIs.

Operation in a soft–real-time environment requires the VHEL to be able to handle deadline overruns. The objective consists in preventing temporary delays from causing a backlog of unprocessed TTIs to accumulate. Clearing this backlog would cause further TTIs to be queued. Older TTIs are unlikely to arrive on time, wasting any processing expended on them. Communication protocol mechanisms remedy the missing information. For

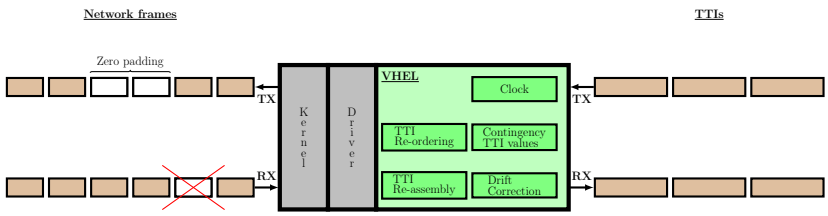


Figure 3.1. VHEL compensation mechanism for late samples.

example, an automatic repeat request mechanism can send a NACK to inform the sender of the loss.

Algorithm 1 presents the VHEL timing recovery logic. When a TTI fails to arrive by its timeout, the VHEL takes action. The timeouts serve to detect two different faults depending on whether the timeout occurred for an RX or TX TTI. RX timeouts result from a loss of connectivity with the frontend. TX timeouts result from BBU processing missing its deadline.

The duration of timeouts in Algorithm 1 is set according to the deadline miss probability. The probability of deadline miss depends on the distribution of total processing time F_{P}. This depends on the BBU processing time T_{BS}, OS delays T_{OS} and fronthaul transmission latency T_{FH}.

$$F_{\mathrm{P}}(T_{\mathrm{BS}}, T_{\mathrm{OS}}, T_{\mathrm{FH}}) \leq P_{\mathrm{out}}. \tag{3.1}$$

Knowing the distribution of F_{P} one finds the probability that the TTI will be late. When making decisions, the VHEL compares the estimated probability to the target threshold P_{out}. The processing delay model is created using delay statistics collected by the VHEL to assess average time budget use. Adaptation of processing load can then be performed, assuming that OS and fronthaul delays remain constant. Measuring the performance of a C-RAN platform using late TTIs benefits from being independent of the cause (interrupts, OS scheduling, network delays). Using late TTIs as a metric is also more easily comparable between different protocols than absolute time durations due to differing targets.

The VHEL groups frontend sample flows into full-sized TTIs. Figure 3.1 illustrates the grouping of incoming samples from network frames into TTI-sized buffers. Frames containing samples are received by the VHEL continuously. When all the samples constituting a TTI have arrived, or a timeout occurs, the VHEL sends the TTI buffer for processing to the protocol code. Timeouts are tied to a clock. This clock is recovered from the incoming sample flow. The timestamp from RX samples serves as the master clock. The sample handling logic of the VHEL is presented in Algorithm 2.

Timing offsets caused by imperfect synchronization are compensated by the VHEL. Offsets accumulate as a result of the LO at the receiver and transmitter having slightly different frequencies due to manufacturing

Algorithm 1 Timing recovery algorithm for late TTIs

```
function TX
    while transmitter enabled do
        wait for TTI or deadline                          ▷ from protocol stack

        if TTI ready then
            if timestamp > deadline then
                send TTI                                  ▷ Normal case
            else
                drop TTI          ▷ Contingency TTI already sent at deadline
            end if
        else
            send contingency TTI                          ▷ Maintain timing
        end if
    end while
end function

function IsRXTTIReady(reorderingBuffer)
    if not gap in reorderingBuffer and not reorderingBuffer empty then
        return true
    else
        return false
    end if
end function

function RX
    while true do
        wait for TTI or timeout

        if timeout then
            abort                          ▷ Frontend communication problem
        end if

        if TTITimestamp >= expectedRXTimestamp then
            currentTime ← TTITimestamp
            reorderingBuffer ← TTI
            update deadline

            while IsRXTTIReady(reorderingBuffer) do
                push TTI to upper layer
                update expectedRXTimestamp
            end while
        else
            drop TTI                          ▷ Frontend or network issue
        end if
    end while
end function
```

Algorithm 2 VHEL sample handling algorithm. [**PI**]

```
function ISLATE(burst)
    TXTime ← burst metadata

    if TXTime + estTransferDelay > targetTime then
        return true
    else
        return false
    end if
end function

function MAINLOOP
    while true do
        peakLocation ← correlate synchronization signals
        drift ← peakLocation − expectedPeakLocation
        TXDriftCorrection ← drift
        RXDriftCorrection ← drift
        read RXBurst first slot

        if RXDriftCorrection > 0 then                    ▷ TTI starts too late
            discard RXDriftCorrection samples   ▷ Discard from the beginning
        else                                        ▷ TTI starts too early or on time
            for i = 1...RXDriftCorrection do
                prepend RXBurst with 0
            end for
        end if

        repeat
            TXBurst ← next transmit burst
        until ISLATE(TXBurst) == false                   ▷ Discard late TTIs

        if TXBurst is on time then
            TXTime ← burst metadata

            if TXDriftCorrection > 0 then
                TXTime = TXTime − TXDriftCorrection
            else if TXDriftCorrection < 0 then
                for i = 1...TXDriftCorrection do
                    prepend TXBurst with 0
                end for
            end if
        else
            take device specific action
        end if

        read RXBurst second slot
        send RXBurst to upper layers
    end while
end function
```

tolerances. The offset increases by a small amount for each sample. Over time, the difference amounts to an entire sample. When the protocol code detects this, it instructs the VHEL to adjust by $+1$ or -1 samples. The adjustment causes either the insertion of an additional sample or the dropping of one received sample. BBU code sees this as a sudden jump in timing. In other words, BBU instances see a constantly drifting phase adjusted from time to time by an instantaneous, and much larger, adjustment. As BBUs only see discretized data, sub-sample corrections are therefore only possible rotating the phase of each complex sample computationally or by hardware-specific LO adjustment commands.

The VHEL also hides implementation and platform details from the BBU code. Protocol code thus becomes easier to port between execution environments. Figure 3.2 presents four different deployment options. They differ in the manner in which the VHEL communicates with frontends. Option (a) is the classical SDR-on-GPOS model. An application uses a device-specific frontend driver. The latter will then employ OS services, such as a User Datagram Protocol socket, to send frames to and from the frontend device. In this scenario, all data transits through the OS network stack. Option (b) describes a scenario where the frontend driver uses network acceleration to bypass the OS kernel's network stack. Option (c) corresponds to a situation where the frontend follows some standard interface or the VHEL has in-built support for the frontend used. In such a scenario, there is no additional driver layer for samples and control messages to traverse. Option (d) differs from Option (c) in that the VHEL bypasses the kernel and sends frames directly through the NIC. Kernel bypass offers latency reduction at the cost of losing access to OS networking features. For fronthaul use, this drawback has little importance as fronthaul networks are typically single-purpose, closed networks. Consequently, services such as routing, per-user QoS or ACK-enabled protocols are not needed.

In addition to differences in implementation details, the VHEL must also adapt to changes in processing speed. Faster systems enable larger bandwidths to be employed or, alternatively, TTIs to be shorter. Processing speed can also change over time due to power management changing CPU clock frequency.

In Publication I and Publication II, different computer systems are compared in terms of their late TTI performance. Figure 3.3 shows the deadline miss performance of two computer systems running an LTE BBU. The higher performance system is labelled "fast" in the figure. Results are presented for different LTE bandwidths. Additionally, performance was also measured with CPU cores dedicated to the BBU using the cset utility. Deadline misses are reduced significantly by the use of dedicated cores for each bandwidth tested. The VHEL hides such performance variations from the BBU code. Missing TTIs are replaced by dummy data. Sample level timing is maintained with the missing information treated as channel

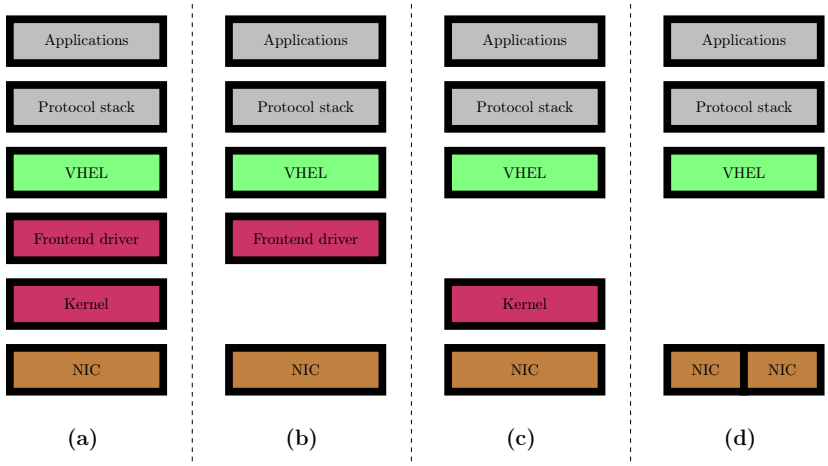


Figure 3.2. VHEL deployment over different networking implementations. (a) OS network stack with frontend driver. (b) Kernel bypass-enabled frontend driver. (c) Built-in VHEL support with OS network stack. (d) Built-in VHEL support with kernel bypass.

fading by the BBU. The VHEL can thus adapt to different hardware configurations and variation in the performance of soft–real-time platforms. Additionally, platform-specific performance management, such as CPU core isolation, can be handled solely by the VHEL. This promotes good software engineering through separation of concerns.

3.4 Architecture and Design

The main task of a C-RAN implementation is to execute RAN code with sufficient performance to enable communication over the air. A suitable architecture accounts for the characteristics of the GPP platform and the protocols to support. Protocols may have differing TTI lengths, ACK delay requirements or other processing time limits. Furthermore, processing load may depend on the node type. Macro and small cells provide different coverage areas, and thus likely different numbers of users served. Smaller cells, with limited coverage, are more likely to encounter situations with no users to serve [27]. UE class impacts processing in a similar manner. Low-power UEs will send data at a lesser rate than MBB high-throughput UEs. Fronthaul latency can also vary and impact available BBU processing time. Load on the fronthaul network depends on the number of active DUs and frontends as well as their functional split [7][96][52]. Higher level functional splits incur a user activity dependent load. BBU processing tasks can thus be scheduled via statistical multiplexing [59] using a processing performance model. Another cause for changes in fronthaul network load is deactivation of unneeded cell sites during periods of low

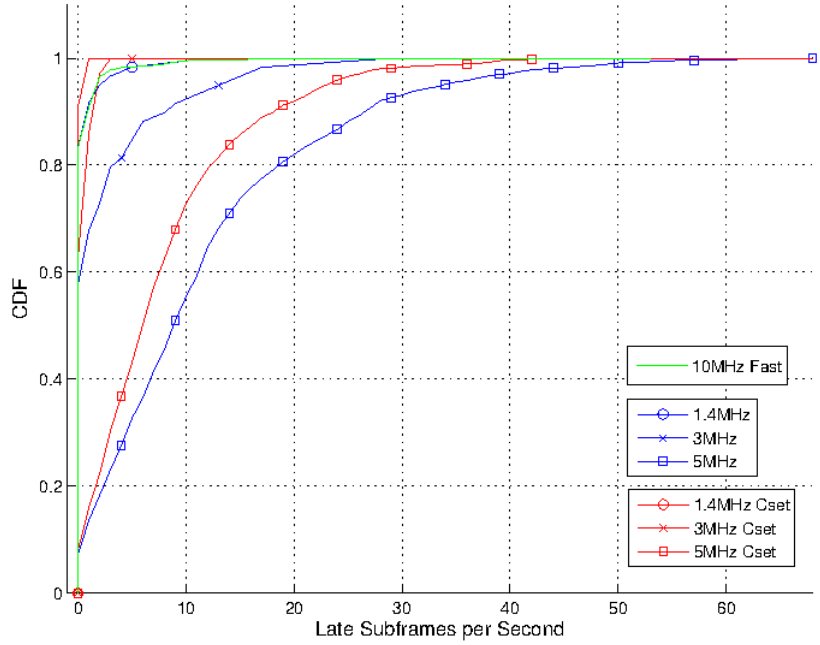


Figure 3.3. Comparison of deadline miss performance between two computer systems and different LTE bandwidths. [**PI**]

demand. In addition to processing time budget tuning, C-RAN platforms must know how long before the desired transmission time samples must be sent to arrive in time at the frontend. It is therefore important for a C-RAN platform to be able to adapt resource usage by performing run time learning and adaption. Optimization can be performed jointly for computation and communication [111]. Applications running on the same platform provide information on their state and receive information on the state of the communication protocol code.

This work focusses on software-based soft–real-time RAN implementations. A characteristic of these implementations is that both load and platform performance can vary over time. Motivated by the many deployment scenarios presented in the literature, a virtualizable and scalable C-RAN architecture was investigated in Publication III. In spite of performance modelling, delay measurements and adaption, deadline overruns can occur. A soft–real-time C-RAN platform must be able to recover from these. Recovery mechanisms aim to restore normal operation as quickly as possible. Doing so requires determining which tasks were late and which other tasks were made late as a consequence. All queued late tasks should be discarded to avoid excessive clustering of deadline misses due to a single event.

Figure 3.4 presents the architecture of Publication III. The lowest layer is the VHEL. It abstracts away hardware and OS specific details from

the communication protocol code. This enables moving instances between virtual machines and hosts even if these have different hardware. TTIs arrive and depart the VHEL via queues. These queues connect the VHEL to a pool of workers. Each worker executes independently of others in its own thread to process either RX or TX TTIs. Each worker gets assigned a TTI to process through the RX or TX work item queue. Work items convey all the information required to perform the task. This design enables workers to be stateless. At the end of processing a TTI, any data needing to be stored are sent to the upper layers. Stateless workers provide several advantages. Firstly, fewer dependencies between threads means they are less sensitive to the ordering selected by the OS scheduler. Secondly, if one worker is delayed, it does not cause TTIs other than its current one to be late. This is because work items are fetched from queues only when the worker is ready to start processing it. Thirdly, additional CPU cores can be better utilized by independent workers. This stems from a lack of need to coordinate access to shared data structures. Finally, because the workers do not store state, some of them may execute on accelerators without shared memory issues. Background workers complement the TX and RX workers to perform long running tasks. These tasks need not complete within the limits imposed by TTI duration.

The highest level of the architecture contains the upper layers of the communication protocol stack. These typically handle computationally lighter tasks than the workers. Upper layers store state and handle resource assignment tasks, such as scheduling. In addition, external interfaces send their messages to the upper layers. A logical system clock provides a time reference to coordinate workers and upper layer functions. This system clock is separate from protocol, OS and sample clocks. The system clock increments monotonically to provide an unambiguous time reference. Upper layers will translate from one time representation to another for external communication. This includes sending and receiving messages between CU and DU. TTIs are identified by a timestamp taken from the system clock. There is thus exactly one TX and one RX TTI with the same system clock timestamp. This enables the VHEL to reorder TTIs. The VHEL also converts from the logical TTI-based system time into sample-based timestamps for use by frontends.

3.5 Power Management

Energy expenditure constitutes a major expense for infrastructure owners [45][14]. Network densification increases the number of cell sites and thus exacerbates the problem [27]. The greater number of cells result in smaller radii and fewer users per cell, on average. Consequently, more cells will be idle. A cell serving no users wastes all the energy it consumes.

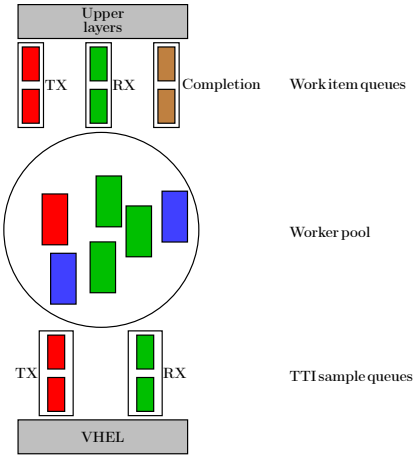


Figure 3.4. Stateless worker architecture.

One prominent approach in the literature consists in switching off lightly loaded cells. Any users they were serving are transferred to other cells. When demand increases, a wake-up procedure be must performed. Both the procedures for initiating the sleep and returning to normal operation generates signalling and introduces latency. These delays may or may not be suitable for UEs depending on their type and use case. Consequently, designing a single policy to suit all users in a cell presents a significant challenge.

The power consumption profile of a C-RAN differs from a conventional network with discrete BSs. Centralization into data centres enables the sharing of costs among BSs in terms of their BBUs. For instance, the whole facility shares cooling and power supply systems. Another difference with regards to discrete BSs lies in the absence of RF components at the C-RAN data centre. The BBU pool's decisions still do affect RF power consumption as RRHs only transmit and receive when instructed to do so by their controlling BBU. By themselves, RRHs make no decisions and maintain no long-term state. UEs are also not explicitly aware of the existence of RRHs. It is thus possible to toggle RRHs on and off rapidly to adapt to varying load conditions. One way to achieve this is to shift load to minimize the number of active BBU tasks [6].

BBU software execution in a C-RAN occurs on a more generic CPU rather than the purpose-built processors of dedicated BSs. Typical CPU power management mechanisms rely on the concept of power management states [40]. States are entered and exited either autonomously by the hardware or under the direction of a power management governor in the OS. Transiting back from a low-power state incurs both a latency and energy consumption penalty. Hence, the decisions of the OS impact the performance of the C-RAN platform. Since the governor in a GPOS is designed to accommodate a wide range of mainly throughput-limited tasks,

its behaviour for latency-critical tasks has shortcomings. Performance can be improved by introducing information sharing between the BS scheduler and OS scheduler similarly to the schedutil [58] CPU frequency governor in the Linux kernel, which operates by sharing information between the task scheduler and the power management subsystem. A similar situation applies to general-purpose hardware platforms, and especially CPUs. Power management for low-latency applications can be enhanced using additional information on the workload [66][16]. The aim is to enable good latency performance without resorting to the traditional solution of disabling low-power states completely [10][46]. Specific low-latency application power management needs specific information, thus the need for a model.

Motivated by the importance of energy consumption identified in the literature, along with the delay constraints, Publication II investigates latency-aware power management for C-RANs. The objective is to provide sufficient performance using the lowest energy consumption achievable. Late TTIs per time unit serve as the performance metric. Late rate varies depending on the frequency of the CPU. A lower CPU clock frequency lengthens processing times and thus increases the likelihood of deadline misses while a greater CPU frequency decreases it. The relationship is, however, not linear. Performing computation faster than required does not provide any advantage. Power management should therefore aim to operate fast enough but no more. Defining the level of performance that is considered fast enough depends on the use-case. Since soft–real-time systems do not, by definition, offer execution time guarantees, there can always be instances of late processing. Late TTIs impact the applications using the RAN as dropped TTIs result in either lost information or additional delays in the case of retransmissions. Measuring the late TTI rate of an implementation consequently provides a metric for the QoS applications experience. A system must therefore be dimensioned for a rate of deadline misses that is compatible with its intended use.

Figures 3.5 and 3.6 show the TTI deadline miss rate as a function of CPU clock frequency. The standard deviation of the deadline miss rate is also shown as vertical bars. As can be seen from Figures 3.5 and 3.6, past a certain CPU frequency, little improvement can be obtained. Such curves of performance per frequency units can be used to build a model of system performance. The results in Figures 3.5 and 3.6 also show that adding load improves performance in terms of variance at higher clock rates. This is likely due to the more constant load preventing the OS scheduler putting the tasks to sleep which, in turn, reduces the delays associated with waking them up. Similarly, the power management functionality in the CPU may stay in a higher performance state in the presence of a constant load.

Publication II uses least-squares curve fitting to produce a predictive model from the recorded empirical data. A method for mapping mea-

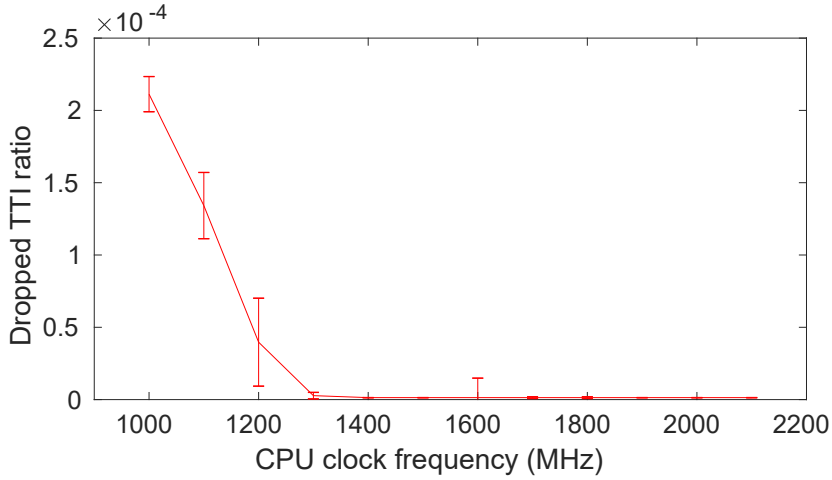


Figure 3.5. Deadline miss rate as a function of CPU frequency with no data plane load. [**P**II]

surements to a model is important, as each system possesses its own performance characteristics. The deadline miss ratio can be modelled as a function of the CPU frequency f and per-system coefficients $\boldsymbol{c}$ in the form

$$F_0(\boldsymbol{c}) = \left(\frac{g_1(\boldsymbol{c}, f, R_\mathrm{s})}{g_2(\boldsymbol{c}, f, R_\mathrm{s})} \right)^{g_3(\boldsymbol{c}, f, R_\mathrm{s})} \tag{3.2}$$

Here, functions g_i are functions of the sampling rate R_s and the CPU frequency. These two factors are weighted by the system specific coefficients $\boldsymbol{c}$. In a practical system, the coefficients would be learned by a background task. Periodically, updated values would be transferred to the operative logic making decisions on a TTI-by-TTI basis. Figure 3.7 compares the empirical and modelled performance for various LTE bandwidths. Higher bandwidths increase the load on the system as the quantity of data to process increases. It can be seen that the model adapts to different loads well, with the largest error being 0.1 normalized late TTIs. The ability to respond to changes in load is important as it allows dealing with variation in soft–real-time platform performance.

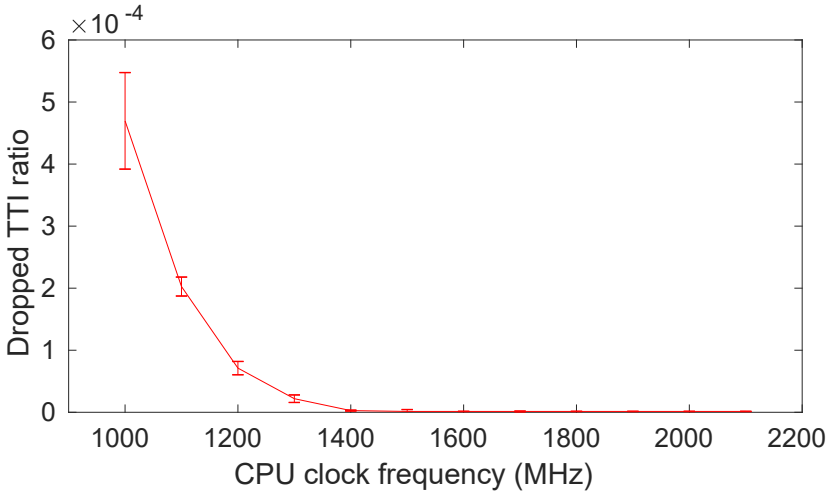


Figure 3.6. Deadline miss rate as a function of CPU frequency with data plane load. [**P**II]

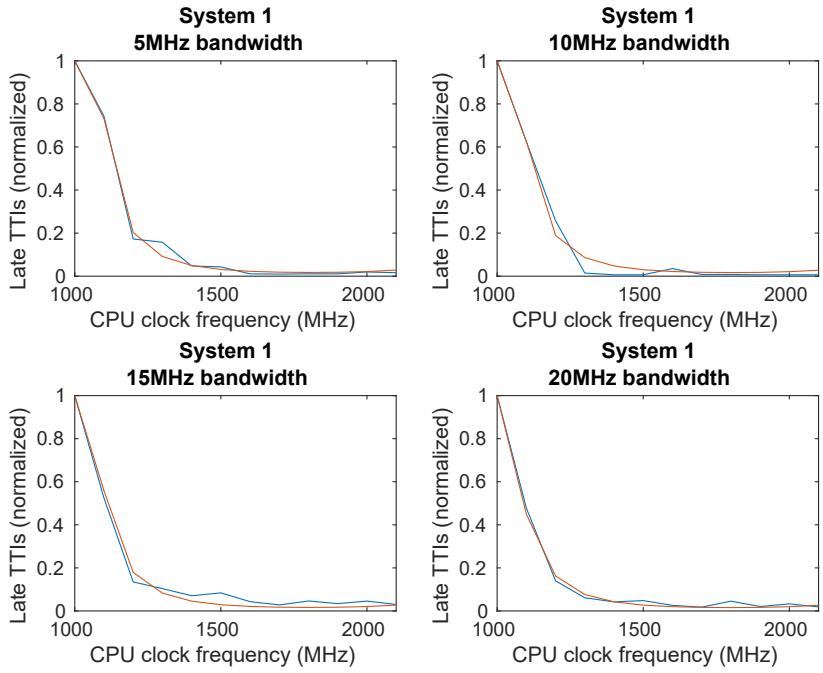


Figure 3.7. Modelled deadline miss rates compared to empirically recorded values. [**P**II]

4. Disaggregated Radio Access Networks

RAN architectures evolve to meet emerging needs. Increased demand, combined with more varied service requirements, drive the densification of RANs using heterogeneous BS types. Implementing this evolution using traditional designs presents challenges in terms of cost, energy consumption and inter-BS co-ordination. This chapter considers RAN disaggregation as a potential solution. This approach decomposes RAN functionality into separate physical nodes, each responsible for a portion the processing required. Processing benefiting from information sharing is placed into a centralized controller. The remaining functionality runs in more energy efficient, cheaper units. The aim is to improve the QoS offered to UEs through increased coverage and capacity while mitigating the challenges of densification co-ordination.

This chapter first reviews prominent new RAN architectures proposed in the literature. Their requirements are identified. The rest of this chapter then presents the contributions of this work towards meeting those requirements. Principles put forth are illustrated using UE positioning as a test application.

4.1 Architectural Evolution

Increased demand, new use cases and technological developments drive network architecture evolution. Data traffic growth has led networks to densify in an effort to provide the requisite capacity. The resulting increase in CAPEX puts pressure on infrastructure owners to seek out lower cost deployment models [43][99]. C-RAN offers an attractive proposition through its ability to centralize BBU processing enabling simpler, cheaper RRHs. Smaller RF units reduce expenditure not only in unit costs but also ancillary costs such as site rent, cooling and power supply. In parallel to cost reduction, new network architectures also introduce the possibility of offering new services. Lower latency combined with more even coverage enables provision of ultra-reliable low-latency communication services.

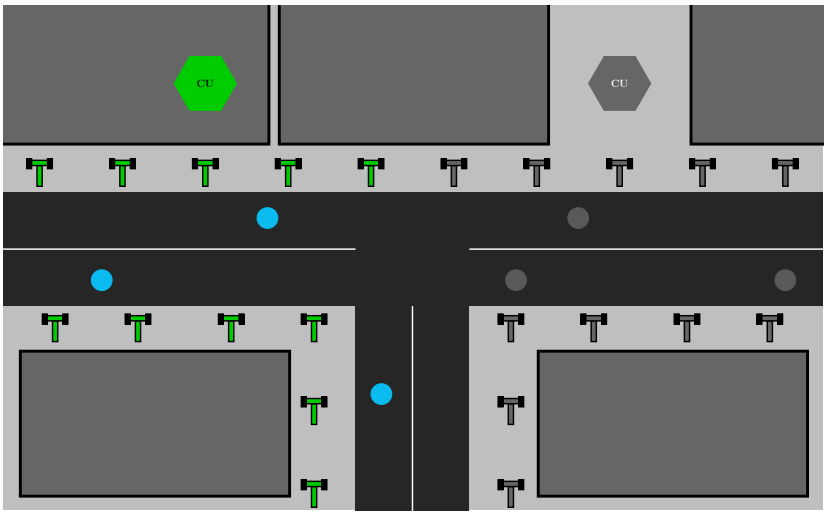


Figure 4.1. Disaggregated network architecture. RF frontends are mounted on lampposts along the street. UEs are shown as circles. The colouring of frontends and UEs indicates their controlling CU.

New service types introduce UEs with usage profiles differing from traditional ones. In particular, machine-type communication applications can behave in markedly different ways compared to human-centric ones. Some application also require computational resources closer to terminals for latency, privacy or data volume reasons [69]. C-RANs and application software can co-exist in edge clouds. Data can then be processed as soon as it is received by the network. Doing so allows only processed data to be forwarded to more centralized clouds. Meeting diversified usage scenarios using a single approach presents challenges. Instead, networks require self-configuration capability to effectively address new, heterogeneous use cases as well as increased network complexity.

Figure 4.1 illustrates a possible deployment option of a next-generation network in a dense urban setting. In this example, the RF frontends are mounted on lampposts. The frontends may be physically separate RUs or they can be integrated into the DU. In the integrated case, the DU itself will be mounted on, or near, the lamppost. UEs are represented by circles. Frontends and UEs are coloured green or gray according to the CU responsible for them. CUs oversee some number of DUs assigned to them. UEs can be served using any of the DUs a CU controls. Switching from one transmission point to the next can be done for multiple reasons such as mobility, load balancing or to improve channel conditions. Since long-term state is only maintained between CU and UE, DU assignments can be updated quickly and often. Multiple DUs can be used to serve a single UE. One use case for this is to provide additional reliability by increasing spatial diversity around an obstacle, such as a tall vehicle on the road.

From a technical standpoint, achieving the objectives of next-generation

networks poses several technical challenges. Firstly, scaling of C-RANs must be understood. In particular, building BBU pools only constitute an economically desirable option if a sufficient number of BBU instances can be co-located. Placing many BBU instances on a single platform forms a key factor of scalability. Secondly, fronthaul and midhaul traffic volumes can be very large. Moreover, this traffic is delay intolerant. Data volumes can be reduced by using higher-level functional splits at the cost of reduced co-ordination benefits. CU-DU-RU communication performance informs the choice of the trade-off between co-ordination benefits and traffic load. Thirdly, mobility and state management poses questions with regards to task distribution. Centralizing all decision making into a single node maximizes the potential for co-ordination at the expense of greater, and potential unacceptable, latency. Distributing decision making enables a reduction in delays but requires distributed management of state.

Solving the above-mentioned challenges requires an understanding of C-RAN system behaviour. Models for latency and jitter enable controllers to make better allocations. The most latency-sensitive applications can be placed in nodes closest to the UEs they interact with. Admission control can also take into account the ability to enforce delay limits in addition to considering capacity.

4.1.1 User-centric Networks

User-centric networks (UCN) aim to improve UE coverage uniformity and reliability [109][70][19]. Instead of UEs adapting to the cell structure of the networks, UCNs adapt to the movements of UEs by building a virtual cell around it. Virtual cells adapt as UEs move with transmission points being added and removed. Maintaining a group of multiple physical transmission sites improves average channel conditions between the RAN and the UE. UCNs thus help improve coverage quality as there are no cell edge users. Virtual cell updates do not cause service interruptions provided at least one cell site remains available to provide data service. UCNs hence provide a significant improvement over traditional schemes with dense networks and beamforming. Network densification reduces cell size, and thus dwell time, while beamforming increases the number of dimensions in the handover decision problem.

C-RANs offer advantages for building UCNs. The co-location of BBU instances enables low latency and overhead in modifying virtual cell configurations. This, in turn, enables more frequent updates than what conventional handovers would enable, resulting in more consistent service quality for UEs. A centralized controller in the C-RAN is further able to consider the load situation of each cell. In case of a MEC-based application, the controller and application may also exchange information. Such information could include predictive load, required QoS and anticipated

mobility patterns.

4.1.2 Ultra-dense Networks

UDNs provide additional capacity through shortening of the average link distance [36][60][20]. Small cell size combined with low numbers of users per cell engenders a situation in which most BS-UE pairs have line-of-sight (LoS) conditions most of the time. The resultant, more predictable channel helps in providing a consistent service to users. Shorter links reduce the needed transmission power, leading to a reduction in energy consumption. On a network level, spatial reuse of frequency increases. Greater area spectral efficiency makes UDNs desirable for infrastructure owners.

UDNs require tight co-ordination to mitigate inter-cell interference and excessive mobility signalling. Various methods have been proposed in the literature [93][107][15] to help alleviate these issues, such as: multi-tier networks, CoMP, predictive mobility management and beamforming. Each scheme offers differing benefits and drawbacks. Scheme selection should consider both network resource availability and UE QoS requirements.

4.1.3 Cell-free Networks

Cell-free networks (CFN) strive to improve the determinism of wireless communication [13][21][79]. The concept uses a large number of geographically distributed antennas under the control of a single central unit. The latter decides which of the distributed antennas to use to serve each UE. Selection from a large number of spatially separated transmitters aims to exploit channel hardening. The large number of potential transmitters increases the likelihood that a short link with favourable propagation condition exists for each UE. It is therefore likely that a suitable group of transmitters can always be found to jointly serve users. This, in turn, promotes uniform QoS throughout the network's coverage area.

Concentrating decision making into a centralized BBU pool enables easier sharing of channel state information (CSI) data between different cells. It also becomes possible to utilize a hybrid model where cells of a cellular network act as a cell-free network by jointly serving some UEs. One use case is to provide ultra-reliable communication service to those UEs requiring it. Traditional handovers would vanish as the UE could be served by multiple transmitters simultaneously. The main challenge to effective CFN implementation comes from the need to co-ordinate the operation of a large number of transmitters. The volume of information to be exchanged with the central controller poses scalability challenges.

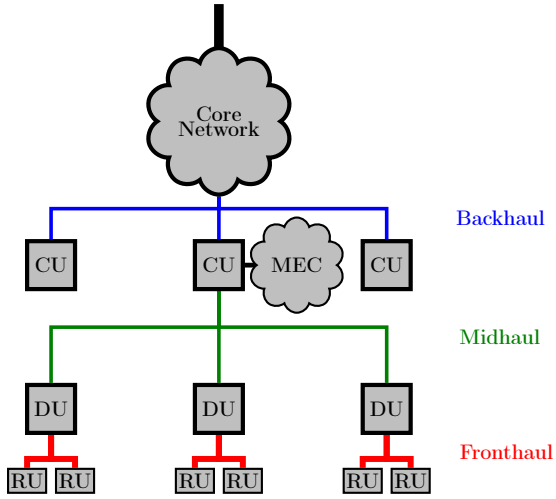


Figure 4.2. Link types between RAN nodes.

4.2 Midhaul

Networks comprise multitudes of nodes to provide wide area coverage, reliability and to support different radio access technologies. Upcoming RAN designs further envision to split the functionality of logical nodes into separate entities linked by a midhaul (Figure 4.2). The midhaul networks linking these nodes together enables their co-operation. Inter-node communication therefore constitutes a critical component of RAN architectures. Even though the midhaul is an interface internal to the RAN, its latency affects the end-to-end latency perceived by UEs. The RAN cannot respond to user requests faster than it can move data among its own components. Furthermore, time spent moving data reduces the time available for computations. Sensitivity of applications to midhaul-induced latency depends on the use case [95]. Placement of RAN component tasks must be considered for each application separately to enable meeting application-specific QoS requirements. This includes the choice of virtualization technology, which impacts delays as data must be moved in and out of virtual instances [37].

Next-generation RANs fully realise their potential only when able to co-ordinate the action of each node. Network designs must therefore consider the impact of the communication links between the various nodes of the RAN. Different approaches, such as UCNs, UDNs and CFNs impose different requirements. One of the most significant design choices is the functional split. The placement of each computation task impacts multiple factors. Firstly, all processing tasks placed away from DUs in a centralized location increase latency from the viewpoint of UEs. From the viewpoint of the RAN, however, the latency in inter-node communication decreases. The resulting improvement in the ability of DUs to jointly serve UEs may outweigh the increased latency between UE and centralized

location. Secondly, traffic load depends on the functional split. Lower level splits engender higher loads. Moreover, the relationship between user payload traffic and midhaul traffic changes. For low level splits, data rates are constant. Higher level splits experience rates dependent on user traffic. Lastly, the availability of computation resources varies depending on the type of node in question. Large BBU pools are more likely to have significant processing capacity available due to lesser constraints on power, cooling and size compared to DUs.

In addition to technical requirements, cost factors impose a need to utilize resources efficiently [99][76]. Low-level functional splits required very high data rates from fronthaul links. Fibre optic links capable of carrying these high rates may not be available. Installation of additional capacity to connect each node can be prohibitively costly. Optimizing the placement of tasks and the routing of traffic enables CAPEX and OPEX savings.

The literature provides works comparing the benefits and drawbacks of various functional splits [62][98][75]. The multitude of possible CU-DU separation points in the protocol stack present different characteristics and trade-offs. The main one is between the gains of centralization and the demands imposed on the midhaul. [98] studies the energy consumption benefits of placing more functions in a centralized site. Doing so is found to reduce total energy consumption but requires more transport capacity between the centralized and distributed sites. A similar objective is pursued in [75] where an optimization framework is presented to this end. It considers both link and node power consumption in order to find an optimal baseband function placement between centralized and distributed sites. Two functional splits, one high and one low, are considered. It is found that DUs consume a very similar amount of power in both splits but the CU consumes more in the lower-level split.

In light of the various trade-offs available, work has also been carried out in adapting the functional split dynamically [22][8]. The idea is to benefit from the advantages of each option when applicable instead of settling on a static choice. This requires taking into account, among other things, the mobility of users as this determines how quickly the optimal situation changes. It is found that while the optimal split changes rapidly, the previous choice does not yield poor performance as quickly. A further consideration in selecting a functional split comes from the load caused by computation offloading. Mobiles devices may offload part of the computational load of the applications running on them to the RAN. The functional split determines the midhaul requirements between CU and DU, and thus has an impact on the delays experienced by offloaded tasks.

The functional split of a RAN impacts many aspects of its design [83]. Different scenarios present different requirements and will thus benefit from different functional splits. It is therefore important to study all

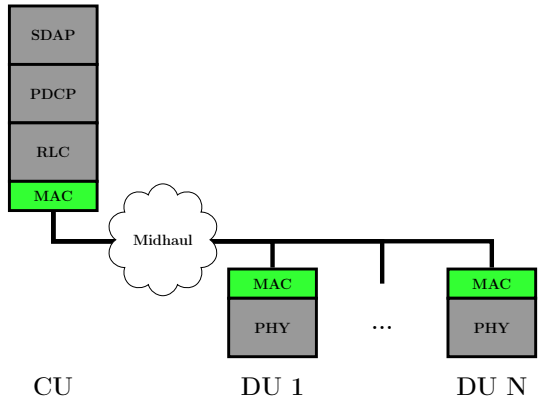


Figure 4.3. Function split considered in this work. [**PV**]

options to provide more tools for adaptation. Operators might also consider further constraints such as MEC placement or network topology [75][91]. Thus the selection of a functional split may be driven by factors beyond its characteristics in terms of air interface design. As result of the multitude of factors to consider, optimal functional split selection remains an open problem [98].

The functional split studied in this thesis is Option 5 [1]. Option 5 is characterized by a compromise in the sense of being a middle ground betwen high- and low-level splits. Its transport requirements are laxer [62] than for lower layer splits but it does not allow for the same level of co-operative processing. Yet, significant gains can still be achieved [88]. It should be noted that while the latency constraints of Option 5 are lesser than for lower-level splits, they remain strict enough to prohibit very long link lengths unless air interface response time degradation is acceptable. One of the main drawbacks of Option 5 is the complexity of the interface between CU and DU. The work in this thesis aims to contribute to the literature by providing testbed-based analysis of Option 5.

In Option 5, RAN functionality is split among CU and DUs at the MAC layer, as shown in Figure 4.3. CUs manage the overall state of the cell but do not operate the air interface used to communicate with UEs. DUs handle this task using one or more radio units (RU) either integrated into the same physical device or as distinct units from DUs. In this work, the question of how each DU is implemented in terms of RU design is not considered in detail. It is assumed that each DU has some suitable and geographically close means of sending and receiving radio frequency (RF) signals to and from UEs.

The CU manages the DUs under its control to serve the geographical area assigned to this CU. Functionalities handled by CUs operate on a slower timescale than the functionalities placed in DUs. Co-operation takes place at the cell-level by means of inter-DU inteference control, centralized pre-

coding decisions and beam direction management. No joint RF processing takes place as this would induce very high signalling traffic load betwen CU and DUs as well as require a complex interface for information sharing. The CU thus serves as a manager of the spectrum assigned to its cell without directly managing PHY processing. Consequently, CoMP is possible at the level of coordinated scheduling while joint transmission CoMP is not possible. The sharing of sufficient channel state information to enable joint processing would greatly increase the complexity and overhead of the implementation. This thesis instead considers a model where each DU operates a logical cell. RUs are assigned to each DU to enable them to provide coverage in the desired service area. For example, RUs, and their associated antennas, could be mounted on one of more neighbouring lampposts along a street. The service area of a DU would therefore be the area covered by its RUs, while a CU would serve an area covered by its DUs.

Instructions are periodically sent to DUs by the CU. For each period, the DUs are provided data to transmit to UEs and parameters related to interference mitigation. DUs then independently perform link adaptation, retransmission and RU selection. At the end of each period, each DU reports to the CU the relevant events and statistics that occurred during the period. This includes, for example, the number of retransmissions and a list of UEs detected. The CU makes use of the reports it receives to make decisions for the next period. The studied model reduces signalling requirements and interface complexity between the CU and DUs compared to joint RF processing, which requires the sharing of channel state information.

Information exchange between logical cells over inter-cell interfaces, such as X2 in LTE, is handled by the CU. It communicates with neighbouring CUs to exchange relevant information when UEs cross from the area of one CU to the next. Information sharing between DUs can take place through two mechanisms. The first option is to have the CU distribute relevant information as it constantly communicates with DUs. The second option is to have DUs pass information to each other directly over the midhaul.

DU instances are grouped together as virtual processes inside a physical server. Grouping DUs together reduces the installation footprint and cost compared to having physically separate DUs. Virtualized DUs share the cost of power supplies, cooling systems and other support costs. Use of the midhaul for inter-DU communication can be eschewed in the case of co-located DUs. In this scenario, the DUs can use the inter-process commmunication services offered by the host OS. Doing so lessens the latency and load incurred by the use of the midhaul. One application scenario where this is beneficial is a UDN deployment. Placing antennas on street furniture is easier than placing fully-fledged base stations. These antennas are connected to the DU server via optical cables. A CU controls multiple such clusters of DUs and their connected antennas to cover the

required service area. Another use-case benefiting from co-located DUs is where a neutral host builds out infrastructure that is then rented out to mobile netword operators. Each operator would then place their own DU instances on the neutral host's server.

Hosting multiple DUs in one server improves the flexibility of service provisioning. Each DU may then be assigned to provide a specific service type using a number of antennas. For example, one DU server could host a DU optimized for eMBB use and another DU would support mMTC use. Multiple RATs can also be supported on the same hardware instead of needing a per-RAT base station. While performing UE-to-DU allocation, the CU selects the appropriate DU to provide the requested service type. DUs and antennas differ in terms of their visibility to UEs. UEs are aware of the DUs in so far as each as each DU operates a particular configuration for its control plane. Antennas, on the other hand, are not visible to UEs as separate logical entities. For instance, one DU might represent an LTE eNB using four antennas sharing the same cell-specific pilots.

The CU can adapt to decreasing load by shutting down low-load DUs and their associated computing resources. Doing so reduces power consumption. Increasing demand can be responded to by doing the converse. Another benefit of grouping DUs into the same server comes from flexibility of antenna assignment. Antennas can be reassigned to another DU. This can be done to switch over in case of a crash, to perform an upgrade or to shift capacity from one service type to another.

4.2.1 Midhaul Performance Modelling

The literature presented in the previous section present methods for designing and dimensioning their proposed architecture. Most works abstract the implemenation away and only consider latency and jitter as the result of network topology. To complement the existing literature, this work uses testbeds to study the behaviour of soft–real-time midhaul-based platforms. Models are developed to assess the impact of implementation choices on overall RAN performance. This is important as a proper understanding of implementation constraints is necessary to make appropriate decision concerning RAN architectures.

In Publication V, measurements quantify the impact of the midhaul on communication delay between CU and DUs. A commodity hardware testbed is used to measure the delay experienced by commands and reports between CU and DU. Reports are used by DUs to update the CU as to the state of UEs and channel conditions. Commands are messages sent by the CU to DUs or UEs. Both DU-to-CU and CU-to-DU messaging are required during normal operation. Measurements are used to construct predictive models. The ability to estimate the midhaul delay is important for network planning and tuning. Larger numbers of DUs per CU increase

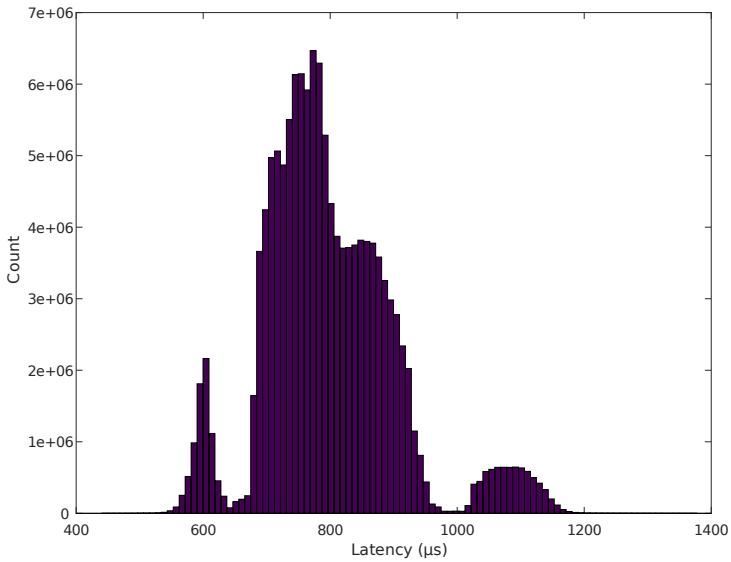


Figure 4.4. Histogram of command latencies from CU to UE.

the opportunity for joint operation. At the same time, latency and jitter also increase. This is because all DUs share the midhaul, which increases the possibility of queuing. In some cases, such as periodic status reports, DUs may all attempt to send a message simultaneously. When this occurs, all but the first one will experience additional delay. To understand the effect of midhaul latency on system performance, it is important to have a model for midhaul latency. To this end, a second order model is introduced, predicting the mean and standard deviation of reporting latency from DU to CU as a function of the number of DUs per CU;

$$f_{\text{mean}}(N_{\text{DU}}) = C_1 * N_{\text{DU}} + C_2 \tag{4.1}$$

$$f_{\text{std}}(N_{\text{DU}}) = C_1 * N_{\text{DU}} + C_2 \tag{4.2}$$

where N_{DU} is the number of DUs, $f_{\text{mean}}(N_{\text{DU}})$ is the mean delay of reporting and $f_{\text{std}}(N_{\text{DU}})$ is the standard deviation of reporting delay.

Soft–real-time C-RANs on a GPP platform experience variability in their processing and communication delays. Figure 4.4 presents a histogram of the distribution of the typical command latency from CU to UE. Larger delays than the maximum of 1.4 ms of the figure values were also observed. These occurred with a probability of 10^{-3}, and were up to an order of magnitude larger than the typical delay.

Midhaul performance is also affected by clock synchronization in addition to latency and jitter. A message can arrive after its intended time because it was sent late rather than due to accruing delay in transit. Communicating

nodes must therefore have a sufficiently similar notion of current time for timestamping and scheduling. The impact of offsets varies per functionality. For example, beam pointing direction updates for fast moving UEs suffer than reports on average DU load. This difference stems from the validity time of the information contained in the message. Average load varies from slowly than the channel of a fast moving user. Similarly, shorter TTIs require more accurate synchronization as a fixed error is proportionally a greater proportion of the TTI duration.

4.2.2 Midhaul Synchronization

A synchronization mechanism is required between the nodes connected to the midhaul. As commodity Ethernet lacks a clock transfer mechanism, synchronization must be implemented using a protocol on top it. PTP provides clock synchronization in local networks. Accuracies of tens of nanoseconds can be obtained in certain circumstances [84][105]. The properties of the network, its load and the characteristics of the nodes all impact the achievable accuracy and jitter. GPP platforms are typically assembled from discrete components communicating over shared interfaces. Communication between the NIC's PTP clock and the CPU can cause jitter in synchronization due to bus contention [25]. Uncertainty in the timeliness of data movement can be compensated by introducing inter-frame gaps at the expense of throughput. In addition, virtualization impacts PTP performance [72]. Timing synchronization experiences an extra delay between host OS and the virtual machine or the container hosting the application.

Synchronization in disaggregated RAN architecture goes beyond sharing a sample clock beyond BBU and frontend. CU and DUs must also agree on a common notion of current time. Predicting GPP-based C-RAN performance is challenging due to the numerous factors affecting the latency of the midhaul and their variability. The separation of the layers of the communication protocol to physically separate nodes introduces distinct clock domains. Furthermore, the heterogeneous nature of the nodes may result in different optimal operating time frames. An additional challenges comes from the use of commodity hardware. Unlike traditional bespoke equipment, commodity hardware is not design to accept an external clock signal. Literature exists on the behaviour of individual components but few works address a complete midhaul-based disaggregated BS. This work address the need to quantify midhaul synchronization performance by analysing end-to-end testbed performance. Models for studying the characteristics and impact of midhaul latency are presented.

The suitability of PTP for C-RAN must be investigated in terms of end-to-end performance for communication applications. Publication V studies the latency performance of a soft–real-time GPP midhaul testbed. Beyond

synchronizing the OS clocks, a C-RAN system must also synchronize the logical time used between CU and DUs. One option for this could be to use TTIs as on the air interface. This method suffers from two drawbacks. Firstly, all DUs must operate at the same air interface configuration, at least in terms of TTI duration. It may be desirable to operate different configurations for different services. Secondly, TTI durations tend to be short leading to tight synchronization requirements between CU and DU even though the CU does not actively control over-the-air transmissions. A better alternative consists in defining a separate logical clock for CU-DU co-ordination. The testbed of Publication V uses such an architecture. Figure 4.5 presents the various clock domains present in the architecture. The CU broadcasts its own logical time on the midhaul. DUs convert this clock domain to the TTI duration of their air interface configuration. In the testbed, synchronization between CU and DUs over the midhaul was implemented using PTP.

Figure 4.6 reports the observed latency for 2, 4, 8 and 16 DUs sharing the midhaul. Delay is here defined as the difference between the expected arrival time for reports and the actual time of receipt. The expected arrival time is set by the CU. Results therefore include both clock offsets and transmission delays. Both jitter and queueing related issues can be seen. The number of peaks reflects the number of DUs, and therefore queueing, while the spread around each peak follows from the variance present in GPP platforms.

In Publication V, the DU reporting latency shown in Figure 4.6 is modelled using a mixture of log-normal distributions. The number of log-normals depends on the number of DUs N_{DU}. The parameters μ and σ of each log-normal are computed as

$$p_{\mathrm{peak}} = 53.625 + \frac{N_{\mathrm{DU}}}{2} + 26.1047n$$

$$\mu = \begin{cases} \log\left(\frac{p_{\mathrm{peak}}^2}{\sqrt{4.25+p_{\mathrm{peak}}^2}}\right), & \text{if } n = 0 \\ \log\left(\frac{p_{\mathrm{peak}}^2}{\sqrt{10+p_{\mathrm{peak}}^2}}\right), & \text{otherwise} \end{cases}$$

$$\sigma = \begin{cases} 0.03, & \text{if } n = 0 \\ \sqrt{\log\left(1+\frac{10+n}{p_{\mathrm{peak}}^2}\right)}, & \text{otherwise} \end{cases}$$

where n is in $[0, N_{\mathrm{DU}}]$.

Assessing the accuracy of a model is important to understand its applicability. Total variation distance (TVD) quantifies the difference between two probability distributions. TVD is calculated as:

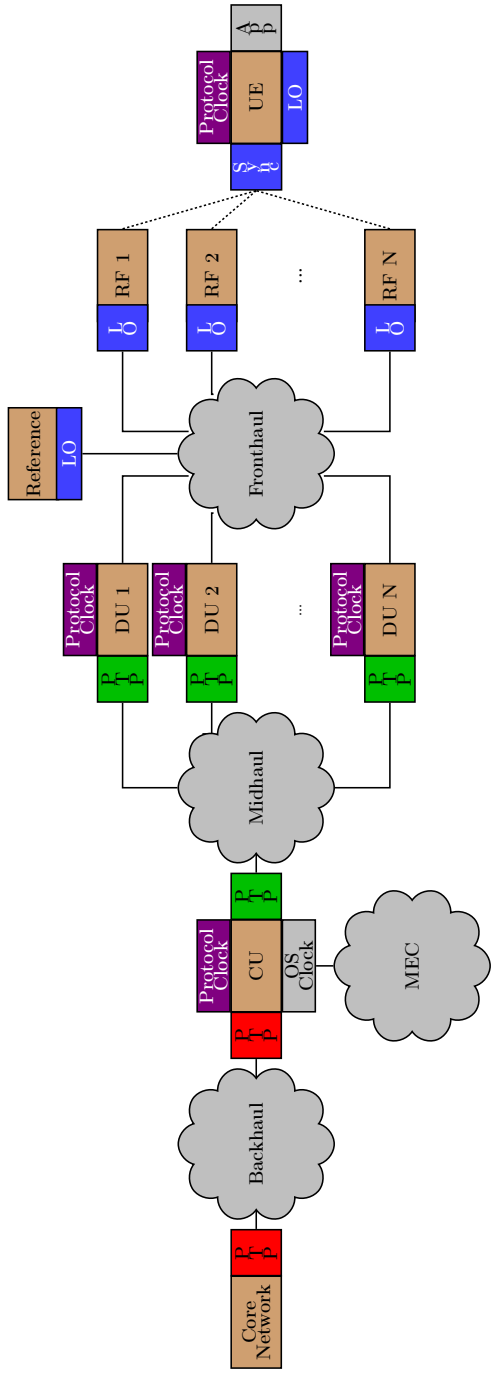
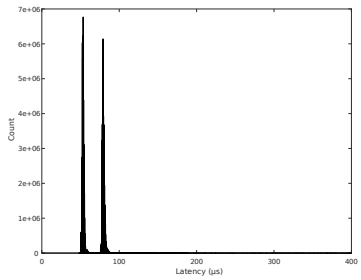
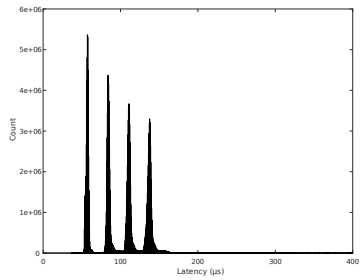


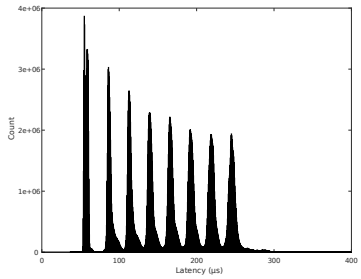
Figure 4.5. Clock domains in a disaggregated RAN.



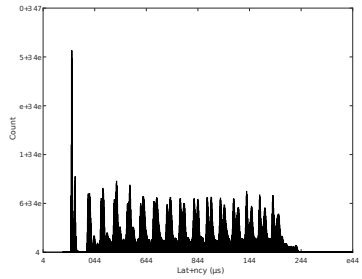
(a) 2 DUs.



(b) 4 DUs.



(c) 8 DUs.



(d) 16 DUs.

Figure 4.6. DU-to-CU report latencies for 2, 4, 8 and 16 DUs. [**P**V]

$$\mathrm{d_{tv}} = \frac{\sum_{x \in X} |p(x) - q(x)|}{2}$$

where X are the latency values being compared, $p(x)$ are the empirical latency bin counts and $q(x)$ are the predicted latency bin counts. A value of zero indicates identical distributions, while a value of one indicates maximally different distributions. The TVD for the DU reporting latency model of Publication V using a bin size of 10 µs is 0.144, 0.160, 0.142, 0.234 for $N_{\mathrm{DU}} = 2, 4, 8, 16$. The low TVD values indicates the model captures the shape of the distribution. Residual differences in predicted latencies are less important as whether most delays are expected to be over or under the TTI processing deadline. Soft–real-time commodity platforms introduce the possibility of variation in timing. The model provides sufficient precision to be used as a tool in estimating the number of DUs that a system can support for a given latency target.

Publication VI considers the impact of midhaul latency on RAN scheduling performance. Deadline misses decrease spectral efficiency as DUs will receive instructions required for transmission to UEs too late. The probability of such wasted transmission opportunities can be reduced by

lengthening TTI durations. However, doing so also reduces the adaptability of the RAN to changing channel conditions and lengthens response times to UE requests.

System behaviour differs significantly depending on how deadline misses appear. Grouped deadline misses will result in infrequent but long service interruptions, which may even cause events of correlated radio link failures across UEs. Publication VI uses a Markov chain to study the distribution of outages as well as their durations. A Markov chain captures memory effects in systems through its pair-wise state transition probabilities. Table 4.1 shows recorded midhaul latencies divided into five classes labelled S_i. Classification is done such that a TTI is put into a class depending on the range within which its midhaul latency falls. Class $S_{\mathrm{i}+1}$ contains higher latencies than class S_i. It can be seen in Table 4.1 that latencies are unlike to jump from small values (S_1 and S_2) to larger ones. When they do, however, there is a tendency to stay at the larger latencies. Algorithm development can benefit from models predicting longer outages.

Let us define the target late rate R_l as the maximum ratio of missed deadlines to the total number of TTIs can support. Scaling in terms of DUs supported in the midhaul depends on the target late rate. Increasing the duration of a TTI reduces the late rate while adding DUs increases it. Figure 4.7 shows the minimum TTI duration required for a given target late rate as a function of the number of DUs. It can be observed that the laxest ($R_\mathrm{l} = 10^{-1}$) and most stringent ($R_\mathrm{l} = 10^{-6}$) target late rates differ in behaviour. At $R_\mathrm{l} = 10^{-1}$, the number of DU has little effect on the supported TTI duration. The increased number of deadline misses remains below the target for each number of DUs tested. Conversely, for $R_\mathrm{l} = 10^{-6}$, the required TTI durations are noticeably longer a function of the number of DUs. The $R_\mathrm{l} = 10^{-6}$ target late rate tolerates so few deadline misses that TTI duration must be considerably increased to absorb added queueing latency introduced by the additional DUs.

Midhaul characteristics affect a RAN in several ways. Large differences between DUs, and across time, change the assumptions underlying system design. Moving UEs between DU instances can result in performance changes with no corresponding change in the radio channel. Algorithms will have a reduced computation time budget due to the midhaul latency taking up a larger portion of the total time available to compute each TTI. Another important design consideration is the placement of MEC and RAN functions on the same server. Sharing the same hardware as DU instances enables low-latency computation-communication joint processing but increases the risk of interference between software processes.

Table 4.1. Markov chain state transition probabilities for four DUs. Latency classes are labelled S_i. [**PVI**]

	S_1	S_2	S_3	S_4	S_5
S_1	0.73	0.27	4.9e-06	4.9e-06	1.7e-07
S_2	0.77	0.23	9.9e-06	7.6e-06	1e-07
S_3	0.26	0.17	0.44	0.13	7e-03
S_4	0.29	0.18	0.26	0.27	9e-03
S_5	0.31	0.19	0.07	0.42	0.01

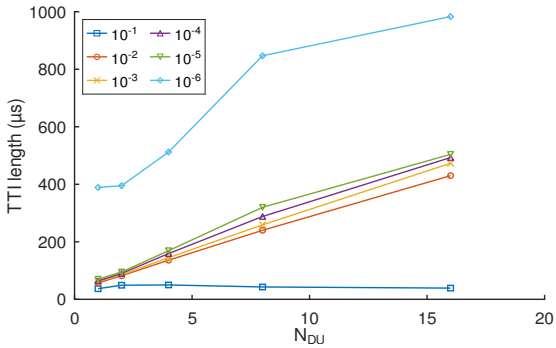


Figure 4.7. Shortest TTI duration supported for various deadline miss rates as a function of DU count. [**PVI**]

4.3 Location-based Mobility Management

C-RAN platforms offer possibilities for the evolution of the RAN. One research direction is the use of accurate positioning in MM. UDNs, UCNs and CFNs can benefit from using a location-based approach instead of the legacy method of UE measurements. Tracking the location of UEs directly spares the need for repeated CSI measurements and reports. In addition to the benefits offered by the reduction in signalling load, RAN localization provides an alternative to global navigation satellite systems. Certain applications require greater precision or increased reliability [29]. In particular, cellular networks can enhance coverage in deep urban canyons where satellite visibility is often poor. Beyond the needs of RAN operation, information on users' mobility patterns can also inform network dimensioning and resource allocation [97].

Implementation of position-based mobility management imposes multiple demanding requirements on the implementation: accurate clock synchronization, low latency, low jitter and management of state across multiple nodes. This becomes especially true when CUs co-ordinate DUs to mitigate interference or users are fast moving. In these scenarios, connectivity may be lost if CUs make incorrect decisions due to stale information.

Publication IV studies the performance of a distributed positioning al-

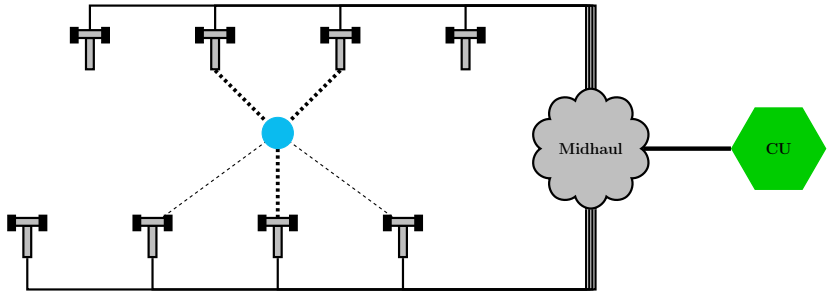


Figure 4.8. Distributed positioning architecture using two extended Kalman filters.

gorithm. Figure 4.8 illustrates the approach used. The implementation splits both the positioning algorithm and state management between CU and DUs. DUs report beacons they can detect along with the AoA of the detection. Additionally, an estimate of the reliability of the detection is also sent to the CU. Weights are applied to emphasise more reliable measurements as part of UE position estimation. Full CSI is not sent from the DUs to the CU. Instead, the DUs handle computation of the AoA using an extended Kalman filter. The results are reported to the CU, which then performs the position estimation using another extended Kalman filter. Algorithm 3 summarizes the main steps.

UE management is split into two levels between CU and DUs, similar to the two-level positioning approach. Together, the CU and DU handle all the tasks required to operate the RAN. Tasks are distributed according to the type of information readily available to each node type. DUs perform functions requiring full CSI, while CUs perform tasks related to managing DUs and UEs.

The two-level approach to UE management confers several advantages. Restricting nodes where UE state information is stored eases UE management as less information needs to be exchanged. Each node maintains only the data it needs to perform its assigned tasks. While full state information is not shared, nodes do exchange values derived from the raw data. These much smaller messages require less transmission and computation resources to send, process and utilize. Splitting state information between CU and DUs implies also decoupling decision making and execution of tasks dependent on that information. In particular, fast and slow scheduling are handled separately. DUs perform fast scheduling since they have access to instantaneous CSI while the CU handles long-term scheduling as it possesses an overall view of all UEs, DUs and backhaul traffic. The CU assigns instructions to each DU to execute in a certain time period, which the DUs then carry out as they see best according to instantaneous channel and UE traffic conditions. Lack of long-term state at DUs provides flexibility in DU-to-UE allocation. The network can thus more responsively adapt to user mobility or changes in service demand.

Algorithm 3 Distributed user-centric positioning algorithm

DUs initializes EKF
DUs begin to transmit synchronization signals
UE attempts to acquire synchronization

if *synchronized* **then**
    UE transmits its beacon

    $AoAEstimate \leftarrow$ EKF ▷ each DU
    $reliabilityEstimate \leftarrow$ channel estimate ▷ each DU
    send $AoAEstimate$ and $reliabilityEstimate$ to CU ▷ each DU

    $database \leftarrow$ DU estimates ▷ from each DU
    $positionEstimate \leftarrow$ EKF ▷ using $database$ values
    send updated instructions ▷ from CU to each affected DU
else
    go back to step 3
end if

Midhaul latency plays a significant part in determining positioning performance in a distributed RAN implementation as the synchronization of DUs occurs over the midhaul using PTP. Use of a GNSS system would remove the impact of the midhaul. Use of GNSS is, however, not always a desirable option due to equipment cost, challenges in indoor reception and dependence on an external network [65][17]. Midhaul performance also impacts positioning accuracy directly. This is because the CU combines the reports of multiple DUs to compute the position of a UE. Delayed DU reports means either degraded positioning accuracy, if reports are ignored, or a delayed computation if reports are waited for. This occurs because the UE continues to move in the meanwhile.

Publication V studies the impact on positioning accuracy of the midhaul in terms of UE position error for various UE speeds. Table 4.2 shows the error in UE positioning caused by midhaul latency for various UE speeds. The DUs are 25 m apart from each other. The UE travels at a constant speed in a line parallel to the line between DUs. A distance of 25 m separates the UE's trajectory to the inter-DU line.

The results in Table 4.2 show that the positioning performance obtained is sufficient for RAN mobility management. The RAN can also provide positioning information to user applications. These application then make their own, separate determination as to the usability of the location information obtained in terms of its precision.

Table 4.2. UE positioning error due midhaul latency and jitter. [**PV**]

User speed ($km\,h^{-1}$)	**Min** (cm)	**Median** (cm)	**99.9 %** (cm)	**99.99 %** (cm)	**99.999 %** (cm)
6	0.09	0.17	0.35	0.45	0.59
30	0.50	0.87	1.74	2.26	2.93
60	0.93	1.75	3.48	4.40	5.53
90	1.48	2.62	5.22	6.80	8.37
120	1.79	3.49	6.98	9.04	11.92
200	3.28	5.82	11.60	14.73	19.15

5. Conclusion and Future Work

C-RANs support the development of improved network architectures to replace legacy hardware-centric networks. Increased use of software in implementations results from the growth in complexity and variety of use cases. Services offered by RANs need to be increasingly tailored to specific application in order for requirements to be fulfilled. Flexible, shared networks offer the possibility of customization without the need for investment in dedicated equipment. Combining centralized BBU pools, midhaul links, heterogeneous RUs and distributed algorithms enables allocation of the most appropriate resources for each user. The successful evolution of RANs into programmable communication fabrics depends on the ability of implementations to meet the requirements imposed upon them. Research must therefore determine suitable techniques to obtain the desired capabilities.

Full exploitation of the flexibility and programmability of software-based C-RAN can only be achieved with an understanding of their characteristics. This work considers software-based C-RAN implementation. The behaviour of various testbeds is assessed to understand their impact on the operating environment of communication systems executing on them. The benefits, requirements and their mapping to the implementation are studied. Approaches to meet these requirements are presented in the areas of midhaul links, distributed state management, scalable software architectures and distributed RAN implementations. The techniques studied aim to aid in the transition from current designs into fully flexible platforms for providing communication services effectively operating as programmable digital-to-RF conversion fabrics.

This work consists of two larger parts addressing two related issues. The first focusses on the implementation of an SDR node using commodity hardware. The second part considers the organization of nodes into a network and the RAN architectures that can be realised in this way. Publication I presents the VHEL concept for hiding the non-idealities of soft–real-time platforms from the communication protocol code. Challenges and solutions thereto for dealing with the non-deterministic behaviour of soft–real-time

systems are analysed. Publication II investigates latency-aware power management. While energy expenditure forms a significant OPEX item, turning devices off presents great challenges in time-division duplex systems. Managing the operating frequency of CPUs provides an alternative less sensitive to nodes being unavailable when they are needed. Publication III and Publication IV study the impact of C-RAN implementation characteristics on positioning performance in a UDN.

Algorithms must also be adapted for distributed execution in the context of C-RANs. Performance imperfections form part of the execution environment. Nodes participating in a distributed algorithm do not share precisely the same notion of current time. Imperfect synchronization impacts not only the C-RAN itself but also algorithms running on it. Any joint decision-making mechanism will encounter uncertainty as to the exact time a measurement has been made. Additionally, the uncertainty varies over time and is thus difficult to quantify and compensate for. Midhaul latency and scaling issues are addressed in Publication V and Publication VI. Distributed measurements combined with centralized decision-making greatly increases the need for time-critical messaging between nodes. C-RAN scaling thus depends on the ability of the implementation to co-ordinate large numbers of entities. The overall approach adopted in this work emphasises compensating errors rather than being overly conservative an effort to offer guarantees. Flexibility in configuration can then be used to adapt RAN services for each use case separately. Such an approach avoids wasting resources by always following the most stringent need or vastly overcommitting resources. Applications receive sufficient QoS for their needs while managing occasional degradations in a manner they deem suitable.

Applying the principles presented in this work has enabled testbeds to show the feasibility of using commodity hardware for implementing RANs. Future work is needed to understand the ramifications on end-to-end performance of multiple simultaneously executing applications. Using the programmability of the C-RAN and information provided by the applications would open up the possibility of more accurately responding, and even anticipating, the communication needs of users. In the same vein, machine learning could provide a powerful tool to exploit data collected by the C-RAN to tune its parameters on a UE-by-UE basis. Another possible research direction is to investigate dynamic task and resource allocation. For instance, an application requiring high reliability could have its processing executed by two independent workers. If one is late due to a missed deadline, the second may still complete in time. Conversely, applications with lax resources could be aggregated to a single worker to exploit statistical multiplexing. Further studies will likely be also needed as technology evolves. In particular, the current trend towards disaggregation of functionality in servers will impact the design of C-RAN platforms. Increased flexibility in configuration and more distributed

designs open up new possibilities to tailor commodity hardware to the needs of C-RAN platforms without losing the benefits of economies of scale.

References

[1] 3GPP. Study on new radio access technology: Radio access architecture and interfaces (Release 14), 2017.

[2] 3GPP. NG-RAN; architecture description technical specification, v. 15.9.0, 2020.

[3] 3GPP. Study on scenarios and requirements for next generation access technologies, v. 16.0.0, 2020.

[4] D. Abbott. *Linux for embedded and real-time applications*. Elsevier Science & Technology, 2017.

[5] R. Agrawal, A. Bedekar, S. Kalyanasundaram, T. Kolding, H. Kroener, and V. Ram. Architecture principles for cloud RAN. In *IEEE Vehicular Technology Conference (VTC Spring)*, pages 1–5, 2016.

[6] W. Al-Zubaedi and H. S. Al-Raweshidy. A parameterized and optimized BBU pool virtualization power model for C-RAN architecture. In *IEEE International Conference on Smart Technologies*, pages 38–43, 2017.

[7] A. Alameer and A. Sezgin. Optimization framework for baseband functionality splitting in C-RAN. In *IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 1–5, 2017.

[8] A. M. Alba, S. Janardhanan, and W. Kellerer. Dynamics of the flexible functional split selection in 5G networks. In *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, pages 1–6, 2020.

[9] O-RAN Alliance. O-ran alliance. `https://www.o-ran.org/`. Accessed on: 2021-05-19.

[10] AMD. Performance Tuning Guidelines for Low Latency Response on AMD EPYCTM 7001-Based Servers - Application Note. http://developer.amd.com/wp-content/resources/56263-Performance-Tuning-Guidelines-PUB.pdf. Accessed on: Sep. 9, 2021.

[11] P. Assimakopoulos, G. S. Birring, and N. J. Gomes. Effects of contention and delay in a switched Ethernet evolved fronthaul for future Cloud-RAN applications. In *European Conference on Optical Communication (ECOC)*, pages 1–3, 2017.

[12] S. Bhattacharjee, R. Schmidt, K. Katsalis, C. Chang, T. Bauschert, and N. Nikaein. Time-sensitive networking for 5G fronthaul networks. In *IEEE International Conference on Communications (ICC)*, pages 1–7, 2020.

[13] E. Björnson and L. Sanguinetti. Cell-free versus cellular massive MIMO: What processing is needed for cell-free to win? In *IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 1–5, 2019.

[14] H. Bogucka and A. Conti. Degrees of freedom for energy savings in practical adaptive wireless systems. *IEEE Communications Magazine*, pages 38–45, 2011.

[15] R. Cai, W. Zhang, and P. C. Ching. Cost-efficient optimization of base station densities for multitier heterogeneous cellular networks. *IEEE Transactions on Wireless Communications*, pages 2381–2393, 2016.

[16] M. Catena, C. Macdonald, and N. Tonellotto. Load-sensitive CPU power management for web search engines. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, page 751–754, 2015.

[17] A. Checko, A. C. Juul, H. L. Christiansen, and M. S. Berger. Synchronization challenges in packet-based Cloud-RAN fronthaul for mobile networks. In *International Conference on Communication Workshop (ICCW)*, 2015.

[18] S. Chen. *Mobility management principle, technology and applications*. Signals and Communication Technology. Springer Berlin Heidelberg, 1st edition, 2016.

[19] S. Chen, F. Qin, B. Hu, X. Li, and Z. Chen. User-centric ultra-dense networks for 5G: challenges, methodologies, and directions. *IEEE Wireless Communications*, pages 78–85, 2016.

[20] S. Chen, Xiang Ji, C. Xing, Z. Fei, and Hualei Wang. System-level performance evaluation of ultra-dense networks for 5G. In *TENCON IEEE Region 10 Conference*, pages 1–4, 2015.

[21] S. Chen, J. Zhang, E. Björnson, J. Zhang, and B. Ai. Structured massive access for scalable cell-free massive MIMO systems. *IEEE Journal on Selected Areas in Communications*, pages 1086–1100, 2020.

[22] Z. Cheng, Y. Tang, and H. Wu. Joint task offloading and flexible functional split in 5G radio access network. In *2019 International Conference on Information Networking (ICOIN)*, pages 114–119, 2019.

[23] F. Civerchia, K. Kondepu, F. Giannone, S. Doddikrinda, P. Castoldi, and L. Valcarenghi. Encapsulation techniques and traffic characterisation of an ethernet-based 5G fronthaul. In *International Conference on Transparent Optical Networks (ICTON)*, pages 1–5, 2018.

[24] Open RAN Policy Coalition. Open ran policy coalition. `https://www.openranpolicy.org/`. Accessed on: 2021-05-19.

[25] J. Coleman, S. Almalih, A. Slota, and Y. Lee. Emerging COTS architecture support for real-time TSN ethernet. In *ACM Symposium on Applied Computing*, pages 258—-265, 2019.

[26] CPRI Initiative. CPRI specification, v. 7.0, 2015.

[27] M. J. Daas, M. Jubran, and M. Hussein. Energy management framework for 5G ultra-dense networks using graph theory. *IEEE Access*, pages 175313–175323, 2019.

[28] E. Dahlman, S. Parkvall, and J. Skold. *4G: LTE/LTE-Advanced for mobile broadband*. Elsevier Science & Technology, 1st edition, 2013.

[29] J. A. del Peral-Rosado, F. Gunnarsson, S. Dwivedi, S. M. Razavi, O. Renaudin, J. A. López-Salcedo, and G. Seco-Granados. Exploitation of 3D city maps for hybrid 5G RTT and GNSS positioning simulations. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 9205–9209, 2020.

[30] M. Einhaus, I. Kim, M. B. Charaf, and P. Arnold. Processing time aware resource allocation in software defined RANs. In *IEEE Conference on Standards for Communications and Networking (CSCN)*, pages 1–6, 2019.

[31] Eurecom. OpenAirSystemRequirements. https://gitlab.eurecom.fr/oai/openairinterface5g/-/wikis/OpenAirSystemRequirements. Accessed on: Sep. 5, 2021.

[32] C. Fan, Y. J. Zhang, and X. Yuan. Advances and challenges toward a scalable cloud radio access network. *IEEE Communications Magazine*, pages 29–35, 2016.

[33] P. Fan, J. Zhao, and C. I. 5G high mobility wireless communications: Challenges and solutions. *China Communications*, pages 1–13, 2016.

[34] H. Farooq, A. Asghar, and A. Imran. Mobility prediction empowered proactive energy saving framework for 5G ultra-dense hetnets. In *IEEE Global Communications Conference (GLOBECOM)*, pages 1–7, 2018.

[35] S. Fernandes and A. Karmouch. Vertical mobility management architectures in wireless networks: A comprehensive survey and future directions. *IEEE Communications Surveys & Tutorials*, pages 45–63, 2012.

[36] M. Filo, C. H. Foh, S. Vahid, and R. Tafazolli. Performance analysis of ultra-dense networks with regularly deployed base stations. *IEEE Transactions on Wireless Communications*, pages 3530–3545, 2020.

[37] F. Giannone, K. Kondepu, H. Gupta, F. Civerchia, P. Castoldi, A. Franklin, and L. Valcarenghi. Impact of virtualization technologies on virtualized RAN midhaul latency budget: A quantitative experimental evaluation. *IEEE Communications Letters*, pages 604–607, 2019.

[38] M. Giordani, M. Polese, M. Mezzavilla, S. Rangan, and M. Zorzi. Toward 6G networks: Use cases and technologies. *IEEE Communications Magazine*, pages 55–61, 2020.

[39] J. Gomes, J. A. L. Silva, and M. E. V. Segatto. Reducing the 5G fronthaul traffic with O-RAN. In *SBMO/IEEE International Microwave and Optoelectronics Conference (IMOC)*, pages 1–3, 2019.

[40] C. Gough, I. Steiner, and W. Saunders. *Energy efficient servers : blueprints for data center optimization*. Apress, 1st edition, 2015.

[41] E. Grayver. *Implementing software defined radio*. Springer, 1st edition, 2013.

[42] L. Guangjie, Z. Senjie, Y. Xuebin, L. Fanglan, N. Tin-fook, Z. Sunny, and K. Chen. Architecture of GPP based, scalable, large-scale C-RAN BBU pool. In *IEEE Globecom Workshops*, pages 267–272, 2012.

[43] M. A. Habibi, M. Nasimi, B. Han, and H. D. Schotten. A comprehensive survey of RAN architectures toward 5G mobile communication system. *IEEE Access*, pages 70371–70421, 2019.

[44] W. T. Han and R. Knopp. OpenAirInterface: A pipeline structure for 5G. In *IEEE International Conference on Digital Signal Processing (DSP)*, pages 1–4, 2018.

[45] Z. Hasan, H. Boostanimehr, and V. K. Bhargava. Green cellular networks: A survey, some research issues and challenges. *IEEE Communications Surveys & Tutorials*, pages 524–540, 2011.

[46] Red Hat. Low latency performance tuning for red hat enterprise linux 7. https://access.redhat.com/sites/default/files/attachments/201501-perf-brief-low-latency-tuning-rhel7-v2.1.pdf. Accessed on: Sep. 9, 2021.

[47] S. A. Hoseinitabatabei, A. Mohamed, M. Hassanpour, and R. Tafazolli. The power of mobility prediction in reducing idle-state signaling in cellular systems: A revisit to 4G mobility management. *IEEE Transactions on Wireless Communications*, pages 3346–3360, 2020.

[48] E. Hossain, L. B. Le, and D. Niyato. *Radio resource management in multi-tier cellular wireless networks*. Adaptive and Cognitive Dynamic Systems: Signal Processing, Learning, Communications and Control. Wiley, 1st edition, 2013.

[49] IEEE. IEEE 1588-2008 - standard for a precision clock synchronization protocol for networked measurement and control systems, 2008.

[50] IEEE. IEEE standard for definitions and concepts for dynamic spectrum access: Terminology relating to emerging wireless networks, system functionality, and spectrum management. *IEEE Std 1900.1-2019 (Revision of IEEE Std 1900.1-2008)*, pages 1–78, 2019.

[51] M. A. Imran, S. Ali Raza Zaidi, and M. Z. Shakir. *Access, fronthaul and backhaul networks for 5G & beyond*. The Institution of Engineering and Technology, 1st edition, 2017.

[52] M. Jaber, D. Owens, M. A. Imran, R. Tafazolli, and A. Tukmanov. A joint backhaul and RAN perspective on the benefits of centralised RAN functions. In *IEEE International Conference on Communications Workshops (ICC)*, pages 226–231, 2016.

[53] H. Jianweiand G. Lin. *Wireless network pricing*. Synthesis lectures on communication networks, # 13. Morgan & Claypool, 1st edition, 2013.

[54] R. Jäntti, N. Malm, K. Ruttik, and O. Tirkkonen. A base station and a method thereto, 2016. World Intellectual Property Organization patent number WO2016113469A1.

[55] F. Kaltenberger, G. de Souza, R. Knopp, and H. Wang. The OpenAirInterface 5G New Radio implementation: Current status and roadmap. In *International Workshop on Smart Antennas (ITG)*, pages 1–5, 2019.

[56] F. Kaltenberger, X. Jiang, and R. Knopp. From massive MIMO to C-RAN: The OpenAirInterface 5G testbed. In *Asilomar Conference on Signals, Systems, and Computers*, pages 608–612, 2017.

[57] A. Karandikar. *Mobility management in LTE heterogeneous networks*. Springer Singapore, 1st edition, 2017.

[58] The kernel development community. Linux kernel **schedutil** scaling governor. `https://www.kernel.org/doc/html/latest/admin-guide/pm/cpufreq.html?highlight=schedutil#schedutil`. Accessed on: 2020-10-13.

[59] H. Khedher, S. Hoteit, P. Brown, R. Krishnaswamy, W. Diego, and V. Veque. Processing time evaluation and prediction in Cloud-RAN. In *IEEE International Conference on Communications (ICC)*, pages 1–6, 2019.

[60] G. P. Koudouridis and P. Soldati. Spectrum and network density management in 5G ultra-dense networks. *IEEE Wireless Communications*, pages 30–37, 2017.

[61] P. A. Laplante. *Real-time systems design and analysis*. Wiley, 3rd edition, 2004.

[62] L. M. P. Larsen, A. Checko, and H. L. Christiansen. A survey of the functional splits proposed for 5G mobile crosshaul networks. *IEEE Communications Surveys & Tutorials*, 21(1):146–172, 2019.

[63] C. Lee, M. Lee, J. Wu, and W. Chang. A feasible 5G Cloud-RAN architecture with network slicing functionality. In *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 442–449, 2018.

[64] J. S. Lee, J. Park, J. Choi, and M. Lee. Design of a management plane for 5G open fronthaul interface. In *International Conference on Information and Communication Technology Convergence (ICTC)*, pages 946–948, 2020.

[65] H. Li, L. Han, R. Duan, and G. M. Garner. Analysis of the synchronization requirements of 5g and corresponding solutions. *IEEE Communications Standards Magazine*, 2017.

[66] X. Li, W. Cheng, T. Zhang, J. Xie, F. Ren, and B. Yang. Power efficient high performance packet I/O. In *International Conference on Parallel Processing*. Association for Computing Machinery, 2018.

[67] Y. Li, B. Cao, and C. Wang. Handover schemes in heterogeneous LTE networks: challenges and opportunities. *IEEE Wireless Communications*, pages 112–117, 2016.

[68] J. Lin, C. Lee, and H. Tsao. On the optimization of user-centric energy-efficient C-RAN. In *IEEE International Conference on Communications (ICC)*, pages 1–6, 2016.

[69] H. Liu, F. Eldarrat, H. Alqahtani, A. Reznik, X. de Foy, and Y. Zhang. Mobile edge cloud system: Architectures, challenges, and approaches. *IEEE Systems Journal*, pages 2495–2508, 2018.

[70] Q. Liu, G. Chuai, W. Gao, and K. Zhang. Load-aware user-centric virtual cell design in ultra-dense network. In *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 619–624, 2017.

[71] C. Maiza, H. Rihani, J. M. Rivas, J. Goossens, S. Altmeyer, and R. I. Davis. A survey of timing verification techniques for multi-core real-time systems. *ACM Computing Surveys*, 2019.

[72] Y. Mingwu and H. Zhenlin. An enhanced end-to-end transparent clock mechanism for the kernel-based virtual machines. In *IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS)*, pages 1–5, 2017.

[73] A. R. Mishra. *Fundamentals of network planning and optimisation 2G/3G/4G*. Wiley, 2nd edition, 2018.

[74] F. Moradi, E. Fitzgerald, M. Pióro, and B. Landfeldt. Flexible DRX optimization for LTE and 5G. *IEEE Transactions on Vehicular Technology*, pages 607–621, 2020.

[75] L. M. Moreira Zorello, S. Troia, M. Quagliotti, and G. Maier. Power-aware optimization of baseband-function placement in cloud radio access networks. In *2020 International Conference on Optical Network Design and Modeling (ONDM)*, pages 1–6, 2020.

[76] Y. Nakayama, D. Hisano, T. Tsutsumi, and K. Maruta. Novel C-RAN architecture with PON based midhaul and wireless relay fronthaul. In *IEEE Consumer Communications & Networking Conference (CCNC)*, pages 1–6, 2020.

[77] O. Narmanlioglu and E. Zeydan. New era in shared C-RAN and core network: A case study for efficient RRH usage. In *IEEE International Conference on Communications (ICC)*, pages 1–7, 2017.

[78] J. Navarro-Ortiz, P. Romero-Diaz, S. Sendra, P. Ameigeiras, J. J. Ramos-Munoz, and J. M. Lopez-Soler. A survey on 5G usage scenarios and traffic models. *IEEE Communications Surveys & Tutorials*, pages 905–929, 2020.

[79] H. Q. Ngo, A. Ashikhmin, H. Yang, E. G. Larsson, and T. L. Marzetta. Cell-free massive MIMO versus small cells. *IEEE Transactions on Wireless Communications*, pages 1834–1850, 2017.

[80] N. Nikaein, R. Knopp, F. Kaltenberger, L. Gauthier, C. Bonnet, D. Nussbaum, and R. Ghaddab. Demo: OpenAirInterface: An open LTE network in a PC. In *International Conference on Mobile Computing and Networking*, page 305–308, 2014.

[81] G. Otero Pérez, D. Larrabeiti López, and J. A. Hernández. 5G new radio fronthaul network design for eCPRI-IEEE 802.1CM and extreme latency percentiles. *IEEE Access*, pages 82218–82230, 2019.

[82] A. Outtagarts, L. Roullet, B. Mongazon-Cazavet, and G. Aravinthan. When IT meets telco: RAN as a service. In *IEEE/ACM International Conference on Utility and Cloud Computing (UCC)*, pages 422–423, 2015.

[83] M. Peng, Y. Sun, X. Li, Z. Mao, and C. Wang. Recent advances in cloud radio access networks: System architectures, key techniques, and open issues. *IEEE Communications Surveys & Tutorials*, 18(3):2282–2308, 2016.

[84] M. Pravda, J. Vodrazka, and P. Lafata. Simulations and measurements of packet network synchronization by precision time protocol. In *International Conference on Telecommunications and Signal Processing (TSP)*, pages 116–120, 2012.

[85] R. Qingyang Hu and Y. Qian. *Heterogeneous cellular networks*. Wiley, 1st edition, 2013.

[86] F. Reghenzani, G. Massari, and W. Fornaciari. The real-time Linux kernel: A survey on PREEMPT_RT. *ACM Computing Surveys*, 2019.

[87] V. Q. Rodriguez, F. Guillemin, A. Ferrieux, and L. Thomas. Cloud-RAN functional split for an efficient fronthaul network. In *International Wireless Communications and Mobile Computing (IWCMC)*, pages 245–250, 2020.

[88] Rony Kumer Saha, Shinobu Nanba, and Kosuke Nishimura. Clustering and centralized resource scheduling of 3D in-building small cells for intra MAC functional split control-/user-plane decoupled CRAN. In *2018 IEEE International Conference on Communications (ICC)*, pages 1–7, 2018.

[89] O. M. D. Santos and A. Wellings. Measuring and policing blocking times in real-time systems. *ACM Transactions Embedded Computing Systems*, 2010.

[90] A. Z. Selim, N. E. El-Attar, M. E. Ghoneim, and W. A. Awad. Performance analysis of real-time scheduling algorithms. In *International Conference on Internet Computing for Science and Engineering*, pages 70—-75, 2020.

[91] M. Shehata, A. Elbanna, F. Musumeci, and M. Tornatore. Multiplexing gain and processing savings of 5G radio-access-network functional splits. *IEEE Transactions on Green Communications and Networking*, 2(4):982–991, 2018.

[92] D. D. S. Souza, R. F. Vieira, M. C. D. R. Seruffo, and D. L. Cardoso. A novel heuristic for handover priority in mobile heterogeneous networks. *IEEE Access*, pages 4043–4050, 2020.

[93] Y. Sun, Y. Chang, M. Hu, and T. Zeng. A universal predictive mobility management scheme for urban ultra-dense networks with control/data plane separation. *IEEE Access*, pages 6015–6026, 2017.

[94] D. A. Temesgene, J. Nuñez-Martinez, and P. Dini. Softwarization and optimization for sustainable future mobile networks: A survey. *IEEE Access*, pages 25421–25436, 2017.

[95] Y. Tsukamoto, R. K. Saha, S. Nanba, and K. Nishimura. Experimental evaluation of RAN slicing architecture with flexibly located functional components of base station according to diverse 5G services. *IEEE Access*, pages 76470–76479, 2019.

[96] D. Wang, Y. Wang, S. Meng, and X. Zhang. Game based wireless fronthaul C-RAN baseband function splitting and placement. In *International Conference on Mobile Ad-Hoc and Sensor Networks (MSN)*, pages 400–404, 2016.

[97] N. Wang, X. Wang, P. Palacharla, T. Ikeuchi, and W. Xie. Mobility-aware 5G midhaul network design for optimizing edge computing resources. In *Optical Fiber Communications Conference and Exhibition (OFC)*, pages 1–3, 2019.

[98] X. Wang, A. Alabbasi, and C. Cavdar. Interplay of energy and bandwidth consumption in CRAN with optimal function split. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–6, 2017.

[99] X. Wang, L. Wang, S. E. Elayoubi, A. Conte, B. Mukherjee, and C. Cavdar. Centralize or distribute? a techno-economic study to design a low-cost cloud radio access network. In *IEEE International Conference on Communications (ICC)*, pages 1–7, 2017.

[100] J. Weinman. The nuances of cloud economics. *IEEE Cloud Computing*, pages 88–92, 2014.

[101] G. Wikström, J. Peisa, P. Rugeland, N. Johansson, S. Parkvall, M. Girnyk, G. Mildh, and I. L. Da Silva. Challenges and technologies for 6G. In *6G Wireless Summit (6G SUMMIT)*, pages 1–5, 2020.

[102] J. Wu and P. Fan. A survey on high mobility wireless communications: Challenges, opportunities and solutions. *IEEE Access*, pages 450–476, 2016.

[103] D. Xenakis, N. Passas, L. Merakos, and C. Verikoukis. Mobility management for femtocells in LTE-Advanced: Key aspects and survey of handover decision algorithms. *IEEE Communications Surveys & Tutorials*, pages 64–91, 2014.

[104] Xenomai project contributors. Xenomai. `https://gitlab.denx.de/Xenomai/xenomai/-/wikis/home`. Accessed on: 2020-10-13.

[105] L. Xue. Research for increasing accuracy of IEEE1588 protocol in the VLAN. In *International Symposium on Computational Intelligence and Design*, pages 136–139, 2013.

[106] C. Yang and Y. Shinjo. Obtaining hard real-time performance and rich linux features in a compounded real-time operating system by a partitioning hypervisor. In *ACM International Conference on Virtual Execution Environments*, pages 59—72, 2020.

[107] B. Yu and Z. Hui. UE's trajectory assisted beam adjustment in 5G mmWave cellular network. In *IEEE International Conference on Computer and Communications (ICCC)*, pages 1145–1149, 2017.

[108] S. Zaidi, S. Affes, M. Azzakhmam, C. Despins, K. Zarifi, and P. Zhu. Progressive hybrid greyfield wireless access virtualization with leveraged combining of cloud, fog, and legacy RANs. In *IEEE International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pages 1–7, 2017.

[109] S. Zaidi, O. Ben Smida, S. Affes, U. Vilaipornsawai, L. Zhang, and P. Zhu. User-centric base-station wireless access virtualization for future 5G networks. *IEEE Transactions on Communications*, pages 5190–5202, 2019.

[110] H. Zhang and L. Dai. Mobility prediction: A survey on state-of-the-art schemes and future applications. *IEEE Access*, pages 802–822, 2019.

[111] Q. Zhang, L. Gui, F. Hou, J. Chen, S. Zhu, and F. Tian. Dynamic task offloading and resource allocation for mobile-edge computing in dense cloud RAN. *IEEE Internet of Things Journal*, pages 3282–3299, 2020.

[112] Y. Zhang, F. Barusso, D. Collins, M. Ruffini, and L. A. DaSilva. Dynamic allocation of processing resources in Cloud-RAN for a virtualised 5G mobile network. In *European Signal Processing Conference (EUSIPCO)*, pages 782–786, 2018.

[113] V. Ziegler and S. Yrjolä. 6G indicators of value and performance. In *6G Wireless Summit (6G SUMMIT)*, pages 1–5, 2020.

Aalto-DT 95/2022

ISBN 978-952-64-0867-5 (printed)
ISBN 978-952-64-0868-2 (pdf)
ISSN 1799-4934 (printed)
ISSN 1799-4942 (pdf)

Aalto University
School of Electrical Engineering
Department of Communications and Networking
www.aalto.fi

BUSINESS + ECONOMY

ART + DESIGN + ARCHITECTURE

SCIENCE + TECHNOLOGY

CROSSOVER

DOCTORAL THESES