UNIVERSITY OF TARTU
FACULTY OF SCIENCE AND TECHNOLOGY
Robotics and Computer engineering

Perseverance Munga Ngoy

PHOENIX: Revisiting Cloudlet Development with Recycled Phones.

Master Thesis (30 EAP)

Supervisor (s): Assoc. Prof. Huber Flores
Assoc. Prof. Ulrich Norbisrath

TARTU, 2023

Abstract

This work revisits the idea of re-purposing e-waste (aka old electronics) into cloudlets that can be used as general computing units for several applications, including edge computing, the Internet of Things (IoT), and pervasive computing applications. While the idea has been around for over a decade, the continuous evolution of personal electronics makes it difficult to keep up to date with general guidelines and principles for re-purposing. As a result, we investigate the latest developments in this topic and provide newer insights for recycling deprecated electronics using modern tools and frameworks. To do this, four Nexus 5 phones are adapted with PostmarketOS, a Linux-based mobile OS, and subjected to benchmark tests alongside a Raspberry Pi 4 and a laptop. Results indicate lower performance metrics in mobile phones, accompanied by substantial heat generation during intensive tasks. The study delves into the advantages, disadvantages, and techniques for optimization, including load balancing, task scheduling, parallel computing, cloud offloading, and energy-aware algorithms. Potential applications are explored, emphasizing community empowerment, low-cost microclouds, and disaster response. While mobile phone-based cloudlets show promise, the challenges and limitations must be addressed. The study underscores the environmental benefits of repurposing mobile phones and proposes sustainable practices, including the removal of batteries for a reduced carbon footprint. The thesis also advocates for the educational empowerment of underprivileged students through accessible technologies, highlighting the potential societal impact of this innovative approach to edge computing.

CERCS: T120 Systems engineering, computer technology

Keywords: E-waste, Microcloud, Cloudlets, Mobile cloud computing, Edge computing, Environmental and social benefits.

PHOENIX: Cloudleti uuesti külastamine Arendus taaskasutatud telefonidega

kokkuvõte: See töö käsitleb uuesti ideed e-jäätmete (vana elektroonika) taaskasutamist pilvekomplektides, mida saab kasutada üldiste arvutusüksustena mitme rakenduse jaoks. Selle alla kuuluvad äärearvutid, asjade internet (IoT) ja erinevad andmetöötlusrakendused. Kuigi idee on kasutusel juba üle kümne aasta, raskendab isikliku elektroonika pidev areng üldotstarbeliste juhiste ja põhimõtete kaasaegsena hoidmist. Tulemusena uurime selle teema kaasaegsemaid arenguid ja edastame uuemaid teadmisi aegunud elektroonika ringlussevõtuks, kasutades moodsaid saadaval olevaid tööriistu ja raamistikke. Töö eesmärgi saavutamiseks kohandatakse neli Nexus 5 telefoni Linuxi-põhise mobiilse OS-i PostmarketOS-iga ning neile tehakse võrdlustestid koos Raspberry Pi 4 ja sülearvutiga. Tulemused näitavad mobiiltelefonide madalamaid jõudlusnäitajaid, millega kaasneb intensiivsete ülesannete läbiviimise ajal märkimisväärne soojenemine. Uuringus käsitletakse optimeerimise eeliseid, puudusi ja tehnikaid: sealhulgas koormuse tasakaalustamist, ülesannete ajastamist, paralleelset andmetöötlust, pilve laadimist ja energiateadlikke algoritme. Uuritakse potentsiaalseid rakendusi, rõhutades kogukonna võimaluste arendamist, odavaid mikropilvesid ja katastroofidele reageerimist. Kuigi mobiiltelefonipõhised pilvekomplektid on paljulubavad, tuleb nendes esinevate väljakutsete ja piirangutega tegeleda. Uurimus rõhutab mobiiltelefonide taaskasutamise keskkonnakasu ja pakub välja säästvaid tavasid, sealhulgas akude eemaldamist süsiniku jalajälje vähendamiseks. Lõputöö pooldab ka ebasoodsas olukorras olevate õpilaste hariduse suurendamist juurdepääsetavate tehnoloogiate kaudu, tuues esile võimaliku ühiskondliku mõju selle servtöötluse uuendusliku lähenemise kaudu.

CERCS: T120 Süsteemitehnoloogia, arvutitehnoloogia

Märksõnad: Elektroonikajäätmed, mikropilv, pilved, mobiilne pilvandmetöötlus, äärearvutus, keskkonna- ja sotsiaalhüved.

Contents

List of Figures

List of Tables

Acknowledgements

Firstly, I would like to express my deepest gratitude to my wife, Benedicte Munga, whose unwavering support and encouragement sustained me throughout the challenging journey of completing this thesis. Special appreciation goes to my dedicated thesis supervisors, Assoc. Prof. Huber Flores and Assoc. Prof. Ulrich Norbisrath, whose guidance, expertise, and valuable insights significantly contributed to the development and completion of this work. Secondly, a big thank you to the Distributed and Pervasive Systems Group; I am truly fortunate to have such a supportive network, and their collective impact on my academic and personal growth is immeasurable.

Next, I would like to express my profound gratitude to my parents, brothers, sisters, and friends for their endless support and encouragement throughout the years of my studies. Finally, I dedicate my work to the memory of my father, Floribert Kakudji Ngoy, who always believed in my ability to succeed academically.

Perseverance Munga Ngoy
December, 2023

1

Introduction

Electronic waste (aka e-waste) is the term used to describe discarded electrical or electronic devices that have become obsolete or broken. The ongoing challenge of how best to dispose of used and unwanted electronics is familiar and dates back to the 1970s[1]. However, a lot has changed since then, particularly the number of discarded electronics today. E-waste is a growing global problem, posing severe environmental and human health threats[1]. E-waste contains toxic substances, such as lead, mercury, and cadmium, that can leach into the soil and water or release harmful emissions when burned[1]. E-waste also represents a waste of valuable resources, such as metals, plastics, and rare earth elements, that could be reused or recycled. According to the World Health Organization, in 2019, an estimated 53.6 million tonnes of e-waste were produced globally, but only 17.4% was documented as formally collected and recycled[2].

One of the primary sources of e-waste is the rapid growth of mobile phones, which are often replaced by newer models or discarded when damaged. Mobile phones are ubiquitous and have many features and functions that make them useful for communication, entertainment, and education. However, mobile phones have limited lifespan and durability and are challenging to repair or upgrade. As a result, millions of mobile phones end up in landfills or are exported to developing countries, where they are often processed in unsafe and informal ways[3].

[1]https://www.ewaste1.com/what-is-e-waste/

[2]https://www.who.int/news/item/15-06-2021-soaring-e-waste-affects-the-health-of-millions-of-children-who-warns

[3]https://www.who.int/news-room/fact-sheets/detail/electronic-waste-(e-waste)

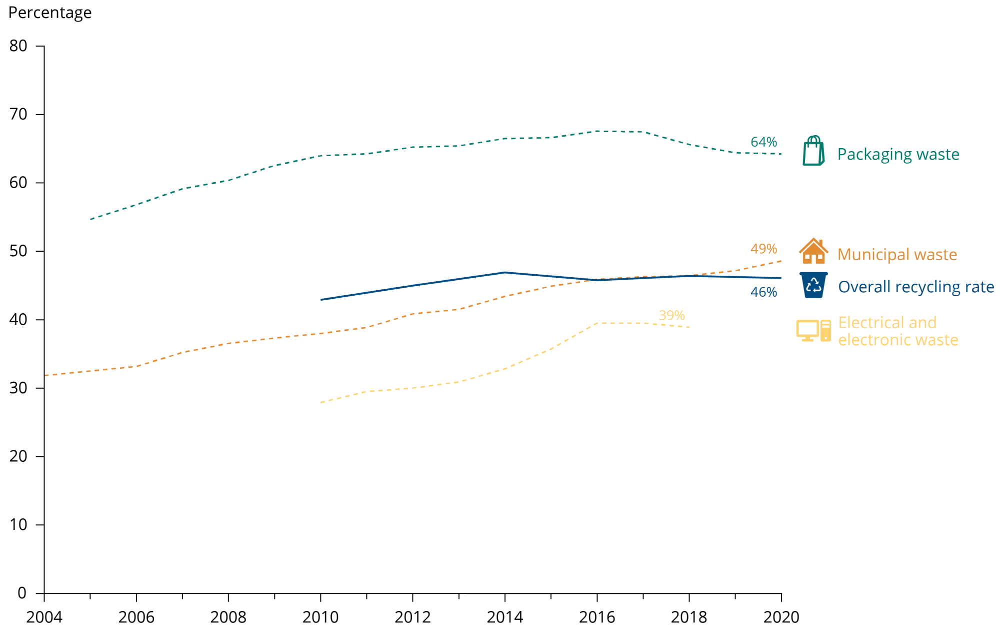

Figure 1.1: Recycling rates in Europe by waste stream (EU-27).

1.1 Waste mobile phone management

Waste mobile phones (WMP) are a significant component of electronic waste (e-waste), one of the world's most rapidly increasing waste streams, due to the explosive demand and shorter product cycles [2]. The circular economy requires understanding consumer behavior regarding mobile phone awareness, consumption, and disposal. Last year alone, there were about 16 billion mobile phones worldwide. Over 5.3 billion mobile phones were destined to be thrown away, according to the International Waste Electrical and Electronic Equipment (WEEE) [3], even though many valuable resources can be recycled from old phones, including copper, silver, palladium, and gold. Many people keep old phones rather than recycle them, and they need to realize that all these insignificant items have much value and, together at a global level, represent massive volumes. Furthermore, the WEEE research shows that a mountain of electrical and electronic waste from phones, tablet computers, toasters, washing machines, and global positioning system (GPS) devices will grow to 74 million tonnes annually by 2030. On the other hand, in Europe, only a few recycling programs are included to tackle these issues, as shown in Figure1.1.

In Australia, where almost all adults have a mobile phone, the only WMP collection and recycling program - "MobileMuster"- operates voluntarily.[4] This study examines how familiar the consumers in Sydney, Australia, are with this program and how they consume and dispose of their mobile phones. The study uses descriptive and statistical analysis (with the Chi-square test of independence and Multinomial Logistic Regression (MLR)) to explore the factors affecting consumer behavior. The findings indicate that only about one-third of the respondents (32.42%) knew about the program and that each household had 3.65 mobile phones on average, with a possession lifespan of 3.17 years (including storage). The findings also reveal that age is positively related to mobile phone ownership and that consumers tend to discard or change their mobile phones even when they are still functional or repairable.

According to the same research[4], the main reasons for disposing of mobile phones are damage or malfunction, lack of additional features, or outdated capacity. The most common behavior after using mobile phones is storing (hoarding) them, especially among respondents aged 18–24 and 25–29, who are less likely to take their WMPs to collection points. The results suggest a need for raising environmental awareness about WMP recycling among consumers.

1.2 General Mobile Phone Repair: Trends and Insights

Mobile phone repair is a growing industry that offers various benefits and opportunities but faces challenges and difficulties. Mobile phone repair can help consumers save money, extend the lifespan of their devices, and reduce electronic waste. It can also create jobs, foster innovation, and support the right-to-repair movement. However, mobile phone repair can be costly, risky, and limited by the availability and quality of repair services and parts. Moreover, mobile phone repair can be difficult and complex, as mobile phones are composed of many small and delicate parts that require special tools, skills, and knowledge. Some mobile phone models are also designed to be complicated or impossible to repair due to the use of glued or soldered components or proprietary or encrypted software. These practices limit the consumers' choices and rights and create barriers and risks for the mobile phone repair industry. According to a study by iFixit, a website that provides repair guides and tools, the average repairability score of smartphones in 2020 was 4.6 out of 10, which means that most smartphones are hard to repair[1]. The study also found that the most common repair issues are cracked screens, battery replacements, and water damage[2].

1.3 Sustainable Mobile Devices: Battery Lifespan, Renewable Energy, and Environmental Impact

Mobile devices' performance, usability, and environmental impact depend on the battery lifespan and sustainable energy. Mobile phones use lithium-ion batteries, which last about two to three years before their capacity drops below 80% [5] and need to be replaced or disposed of. However, this can be costly, risky, and harmful to the environment, as batteries contain toxic substances that can pollute the soil and water or cause fires and explosions[5]. Therefore, sustainable energy for mobile phones, which refers to the use of renewable and low-cost sources of power, such as solar panels, wind turbines, and hydro generators, can be a better alternative. Sustainable energy of mobile phones can save money, extend the battery life, and reduce the carbon footprint of mobile devices[6]. It can also enable mobile devices to work in remote or off-grid areas where access to electricity is limited or unreliable. For example, some projects have developed solar-powered mobile phones, such as the SunCore phone[3], or solar-powered mobile phone chargers, such as the Solio charger[4]. These devices can harness the energy from the sun to charge and power mobile phones without relying on conventional power outlets or batteries.

1.4 Challenges in Accessing Hardware Resource-Constrained Communities

One of the main challenges in resource-constrained communities is the accessibility of educational hardware to students or people interested in the field. Devices such as Raspberry Pi, Jetson Nano, and computers are essential tools for enhancing the quality and accessibility of education in the digital age. However, many less privileged African areas face significant challenges in acquiring, deploying, and maintaining these devices.

[1] https://www.ifixit.com/repairability/smartphone-scores
[2] https://www.ifixit.com/repairability/smartphone-scores
[3] https://suncoresolar.com/
[4] https://www.northernaxcess.com/solio-bolt-solar-charger-battery-back-up

For example, in Africa, only 14 out of 54 countries are where it is possible to find Raspberry Pi-approved resellers[1]. Moreover, the average price of a Raspberry Pi 1 Model A+ device in Africa is 60EUR, which is more than twice the price in Europe (22EUR).

Even if the devices are available in some parts of the continent, they may take a long time to arrive due to the long and complex supply chain and logistics. Moreover, the devices may be damaged or defective due to poor handling or transportation conditions, affecting their functionality and performance. Therefore, the accessibility of the hardware for educational purposes is a significant challenge that needs to be addressed, especially for communities with little resources and access to technology.

This work revisits the idea of using recycled phones as cloudlets. It provides latest insights on the recycling of old electronics as general computing units. It also proposes updated guidelines and principles for reactivating e-waste by using latest available tools and existing frameworks. A cloudlet formed by recycled phones can provide a small-scale and portable cloud data center located at the edge of the Internet, close to the users and devices that need its services. Cloudlets can provide low-latency and high-performance computing resources to support resource-intensive and interactive mobile applications, such as data analysis, image processing, and machine learning. However, building and deploying cloudlets also requires significant investment in hardware, software, and energy, which may be challenging and not affordable for some communities, especially those with limited resources and access to technology. These communities may benefit from the cloudlet services, such as computing and educational services. However, they may not have the means or the opportunity to acquire the devices that enable these services, creating a digital divide and a gap in these communities' development and innovation potential.

1.5 Contributions

The following sums up the contributions:

- **Newer guidelines:** We propose a sustainable way of reactivating e-waste for developing portable cloudlets. This could be helpful in communities with limited resources and could be used for various use cases, such as running and self-

[1] https://www.raspberrypi.com/news/an-update-on-our-work-in-africa/

deploying applications to the Internet and installing a framework used to explore and develop the Internet of Things (IoT). It can be deployed as a low-cost micro-clouds for underwater applications. A practical guide for porting a device (Nexus 5, 2013) using postmarketOS, an open-source Alpine-based Linux distribution is also provided.

- **Updated insights:** To address the issue of batteries and energy consumption, we suggest a standard way of bypassing the Nexus 5 phone's battery to power the phone. Therefore, it provides an excellent opportunity to use sustainable energy to power the phone. We present the limitations of our mobile phone cloudlet, which include the hardware constraints, software compatibility, and energy consumption of the devices, as well as the environmental, social, and ethical implications of reusing e-waste for cloud computing. Furthermore, we discuss how these limitations affect cloudlet services' performance, reliability, and usability and the potential solutions and future work to overcome them.

- **Rigorous evaluation:** A benchmark comparing the performance of the Raspberry Pi and a Nexus 5 phone running Alpine Linux by conducting various tests on CPU performance, memory usage, and disk speed. Besides this, we also evaluate how to integrated cloudlets into a 3D printing rack case, which offers several advantages in terms of deployment and protection of the devices.

Our work addresses the following research questions (RQ):

- RQ1: How can we reduce the weight and size of mobile phones by removing unnecessary components and still maintaining their functionality as cloudlet nodes? This research question is essential because it can help us design and build more portable and easily deployable cloudlets in different locations and environments. It can also help us reduce cloudlets' cost and environmental impact by reusing old or discarded mobile phones.

- RQ2: How can we power the Nexus 5 phone without its battery using alternative energy sources or scavenging ambient energy?
  This research question is essential because it can enable us to extend the lifetime and availability of the phone as a cloudlet node, especially in remote or disaster

areas where power supply is scarce or unreliable. It can also reduce the dependency on batteries, which are costly, bulky, and harmful to the environment.

- RQ3: How feasible is installing Linux-based operating systems on mobile phones and using them as cloudlet nodes?
  Knowing this helps us determine the compatibility and suitability of mobile phones as cloudlet nodes. Linux-based operating systems are widely used in cloud computing and offer many advantages, such as security, stability, and flexibility. By installing Linux-based operating systems on mobile phones, we can enable them to run various applications and services that are compatible with cloud computing.

- RQ4: What are the advantages and disadvantages of using mobile phones as cloudlet nodes compared to other devices, such as Raspberry Pi or laptops?
  The main target of this research question is to help us evaluate the trade-offs and benefits of using mobile phones as cloudlet nodes. Mobile phones have advantages, such as being widely available, having built-in sensors and cameras, and supporting wireless communication. However, they also have disadvantages, such as limited processing power, memory, and battery life. By comparing mobile phones with other devices, such as Raspberry Pi or laptops, we can identify the strengths and weaknesses of each device and choose the best one for our future research work.

- RQ5: How can we optimize mobile phone-based cloudlets' performance, reliability, and energy efficiency for different applications and scenarios?
  This research question is essential because it can help us improve the quality and usability of our cloudlet. Performance, reliability, and energy efficiency are key factors that affect the user experience and satisfaction of cloud computing. By optimizing these factors, we can ensure that our cloudlet can provide fast, reliable, and sustainable services to the users. We can also adapt our cloudlet to different applications and scenarios, such as data analysis, image processing, or machine learning, by tuning the parameters and configurations of our cloudlet.

1.6 Outline

The thesis is structured as follows: Chapter 1 introduces the research background, motivation, objectives, and contributions of the study. Chapter 2 provides the necessary background information and reviews the relevant literature on e-waste, cloud computing, and cloudlets. Chapter 3 describes the methodology, experiments, and design of the proposed Cloudlet using recycled mobile phones. Chapter 4 presents the implementation and evaluation of the Cloudlet and compares its performance with a Raspberry Pi. Chapter 5 discusses Cloudlet's limitations, challenges, and implications and suggests future work directions. Chapter 6 concludes the thesis and summarizes the main findings and recommendations.

2

Background and related work

As explained in the previous Chapter, E-waste is one of the world's fastest-growing and most hazardous waste streams, posing severe threats to human health and the environment. Therefore, finding effective and sustainable ways to recycle devices such as old mobile phones and laptops is a crucial challenge that requires innovative solutions and interdisciplinary approaches. In this Chapter, we review the previous literature on mobile phone and laptop recycling and present a broad overview of different approaches that can be applied to this problem. We focus on three main approaches: micro-clouds (cloudlet), reusing mobile phones, and distributed computing. We explain each approach's concepts, benefits, challenges, and examples and compare them. We also discuss the future research directions and opportunities for mobile phone recycling.

2.1 Reusing Old laptops

Reusing or repurposing old laptops is a way of extending their lifespan and functionality, as well as reducing their environmental impact and resource consumption. Several studies have explored different methods and applications for reusing or repurposing old laptops. The first study[7] proposes a low-cost way of providing laptops to under-privileged students in Electrical and Computer Engineering programs, who may face a digital divide due to the lack of access to computing resources. The paper evaluates the performance of old faculty laptops that are destined for e-waste or surplus sale. It suggests that they can be refurbished and loaned to students who need them. The paper also discusses the benefits and challenges of this proposal, such as improving student learning, reducing environmental impact, enhancing sustainability, and creating a work-study program for laptop repair. The paper concludes that old faculty laptops can offer reasonable performance and usability for engineering software tasks and can be a viable solution to address the digital divide among students.

A second study[8] examines the energy-saving potential of reusing or reselling personal computer devices compared to buying new ones. The paper argues that reusing or reselling old devices can sometimes lead to more energy consumption than buying new ones, especially when the old devices are less efficient than the new ones. The paper analyzes different scenarios of replacing old devices with new ones, such as desktops, laptops, CRT monitors, and LCD monitors. It calculates the energy saving or expenditure for each scenario. The paper concludes that reusing or reselling old devices is not always beneficial for energy saving and depends on the type and efficiency of the devices. The paper also suggests that more investigation and regulation are needed to assess the environmental impacts of reusing or reselling personal computer devices.

In addition to the studies mentioned above, this article[1] proposes a list of suggestions on how to reuse or repurpose old laptops instead of throwing them away or storing them. The article explains the benefits and challenges of each suggestion. The suggestions include:

- Installing Linux, a free and open-source operating system that can run various applications and improve the performance and security of the laptop.
- Installing Chromium, a lightweight and fast operating system that relies on web-based applications and cloud storage, similar to a Chromebook.
- Turning the laptop into a network-attached storage (NAS) system, storing and sharing files and media across different devices and locations.
- Turning the laptop's hard drive into a USB drive, storing and transferring data to other devices.

[1] https://www.cnet.com/tech/computing/how-to-reuse-an-old-laptop/

2.2 Reuse of old mobile phones

Reuse is an essential aspect of sustainability, as it extends the use of old mobile phones by giving or selling them to others [9]. It can also happen when mobile phone manufacturers, retailers, and others resell or donate the phones collected through various schemes. Reuse can improve resource efficiency, reduce dependency on new materials, and provide access to affordable phones for people who cannot buy new ones [10] [11]. However, reuse is only one of many disposal methods adopted by users/consumers. Many store their old phones after replacing them, mainly due to technological obsolescence. It hinders the reuse potential as the value and functionality of the phones decrease over time. Moreover, many users/consumers are unwilling to receive second-hand phones. They are unaware of the benefits and options of reuse. Another barrier is the concern about information security, as users/consumers may worry about their personal data being accessed by others. Therefore, it is necessary to collect the retired phones as soon as possible and to educate the users/consumers about the importance and methods of reuse. It is also helpful to design the phones in a way that facilitates easy disassembly, repair, and upgrading [12] [13]. Most of the previous studies on the disposal of obsolete mobile phones have focused more on recycling than on reuse. Reuse was often a minor aspect or not considered at all. Therefore, more research is needed to explore the factors and solutions that can increase the reuse rate of old phones. Reuse is a key element of the circular economy, and it can contribute to the environmental and economic sustainability of mobile phone consumption.

2.2.1 Consumers' perceptions in mobile phone recycling and re-use

Mobile phones are omnipresent devices that have become essential to our daily lives. One key factor influencing the success of mobile phone recycling and re-use is the consumers' perceptions, attitudes, and behaviors toward these practices. Consumers' perceptions can affect their awareness, willingness, and participation in mobile phone recycling and re-use and their preferences, expectations, and satisfaction with these options.

Several studies have examined consumers' perceptions of recycling and re-use of mobile phones in different countries and contexts. For example, a study was conducted on consumer recycling of mobile phones in the Nordic region[14]. The study explores how environmental messaging can influence consumer decision-making to increase the recycling rate of mobile phones, which have significant environmental impacts and contain valuable resources. The study finds that consumers tend to underestimate the environmental effects of their actions and prefer the do-nothing option, equivalent to disposing of the phone in domestic waste. The study suggests that messaging should be informational, gain-framed, and congruent with the consumer's high-level construal of the issue and that nudge theory can provide some valuable tools for framing the preferred choice. The study also acknowledges some risks and challenges of providing better information to consumers, such as increasing their expectations of receiving payment for waste or exposing the device to dubious recycling practices. The study calls for further research on testing the effects of messaging and understanding consumer motivation and behavior.

Another study conducted particularly in Finland found that consumers' awareness of the importance and existence of waste recovery systems was high, but this did not translate into recycling behavior[15]. The study revealed that 55% of respondents had two or more unused mobile phones at home, and the main reasons for not returning them were lack of knowledge, convenience, or motivation[15]. The study also suggested a need for more information and awareness on mobile phone collection in Finland, especially regarding retailers' take-back[15]. Furthermore, another study in Kenya explored the factors influencing consumers' decisions to replace their mobile phones[16]. The study found that the main driving factors for handset replacement were phone price and functionality, phone brand, battery lifetime, internet connectivity, and phone applications[16]. The study also found that 52.2% of respondents disposed of their retired or damaged phones in regular waste bins, while 34.6% gave them out for additional use by others. The study recommended implementing more awareness campaigns and incentives to promote mobile phone recycling and re-use in Kenya[16].

A fourth study conducted in Australia examined how consumers used, re-used, and recycled their mobile phones and the impact they had on the environment[17]. The study found that consumers were attached to their mobile phones and used them for various purposes, such as communication, entertainment, information, and productivity[17]. The study also found that consumers knew the environmental benefits of recycling mobile phones. However, they were reluctant to part with them due to emotional or functional reasons. The study reported that 40% of respondents had two or more unused mobile phones at home, and only 8% had recycled their old mobile phones in the past year[17]. The study suggested that more education and engagement programs should be developed to encourage consumers to recycle their mobile phones through MobileMuster[18], the only voluntary product stewardship program for mobile phones in Australia.

A fifth study was conducted in Liverpool, UK, exploring the attitudes and perspectives of mobile phone users towards the disposal of their old devices in Liverpool, UK[19]. The study aims to understand the reasons behind the low rates of recycling and re-use of mobile phones and the barriers and opportunities for improving the environmental management of these devices. The study uses semi-structured interviews with 40 participants and analyzes their responses using thematic analysis. The main findings of the study are:

- Most participants have a collection of old mobile phones stored in their homes, usually in drawers, boxes, or bags. They do not dispose of them because they perceive them as valuable, useful, or sentimental objects.
- The participants are generally unaware of the environmental impacts of mobile phone production and disposal and do not consider recycling or re-use as viable options for their old devices. They are also concerned about the security and privacy of their phone data.
- The participants express a lack of trust and information about the existing schemes and facilities for mobile phone recycling and re-use. They are also influenced by their social networks and media exposure in their decisions regarding their old phones.
- The study suggests some recommendations for improving the environmental management of mobile phones, such as raising awareness, providing incentives, enhancing convenience, and fostering social norms.

These studies show that consumers' perceptions of recycling and re-use of mobile phones vary depending on their socio-cultural, economic, and environmental contexts. However, they also reveal some common themes and challenges that need to be addressed by various stakeholders, such as governments, businesses, civil society organizations, and international organizations. Some of these themes and challenges are:

- The gap between awareness and action: Consumers may be aware of the importance and availability of mobile phone recycling and re-use options but may not act on them due to various barriers or constraints. These barriers or constraints may include a lack of knowledge, convenience, motivation, trust, or incentives.
- The tendency to store or dispose: Consumers may store their old or unwanted mobile phones at home or dispose of them in regular waste bins instead of returning them to collection points or giving them out for additional use, therefore resulting in the loss of valuable resources and the increase of environmental pollution.
- The preference for new or updated models: Consumers may prefer to buy new or updated models of mobile phones due to their perceived advantages in terms of features, functionality, performance, or status, leading to frequent replacement cycles and increased e-waste generation.
- The demand for quality and affordability: Consumers may demand high quality and affordability when it comes to recycling or reusing mobile phones. They may expect recycled or refurbished mobile phones to have similar or better features, functionality, performance, warranty, or price than new ones.

2.2.2 Beyond Disposal: Sustainable Alternatives for Unused Mobile Phones

A. Some of the best uses for old iPhones are:

- *Turn it into a VR headset*: We can use our old iPhone to experience virtual reality with a low-cost device like Google Cardboard[1]. We can download VR apps from the App Store or use PlayOnLinux[2], an open-source gaming framework software based on Wine that allows users to easily install Windows-based applications and games on Linux operating systems to run Windows programs on their iPhones. This way, we can enjoy immersive games, movies, and educational content without spending much money on a dedicated VR device.

[1] https://arvr.google.com/cardboard/
[2] https://www.playonlinux.com/en/

- *Turn it into a security camera*: We can use our old iPhone to monitor our home, kids, pets, or anything else remotely. We need to download a security camera app like AlfredCamera[1] on our old and new iPhones and connect them to Wi-Fi. Then, we can mount the old iPhone where we want it and use our new iPhone to view the live feed. We can also get alerts, record videos, and talk through the camera.
- *Turn it into a digital photo frame*: We can use our old iPhone to creatively display our photos and slideshows. We can download a photo frame app like LiveFrame[2] on our old iPhone and choose the photos we want to show from our camera roll or Flickr[3] albums. We can customize the display with filters, transitions, and time settings. We can then put our old iPhone in a frame and place it on our desk, wall, or anywhere we like.
- *Turn it into a smart-home or Apple TV remote*: We can use our old iPhone to control our smart-home devices or Apple TV easily. We need to connect our old iPhones to Wi-Fi and use the Home app or the Apple TV Remote app to access our devices. We can also use Siri to control them with our voice. We can then leave our old iPhone in a convenient location, like our living room or bedroom, and use it as a dedicated remote.
- *Turn it into a scanner and fax machine*: We can use our old iPhone to scan and fax documents without a printer or scanner. We need to download a scanner app like Evernote Scannable[4], on our old iPhone and connect it to Wi-Fi. Then, we can use the camera to scan any document and save it as a PDF or image file. We can also use the fax app to send and receive faxes to and from 180 countries for free.

B. Turn Your Old Android Device into a Useful Gadget

- *Wireless trackpad and controller for a computer*: One way to use an old Android device is to use it as a wireless trackpad and controller for a computer. We can do this by installing an app called Unified Remote on both our old

[1] https://alfred.camera/
[2] https://liveframeapp.com/
[3] https://www.flickr.com/
[4] https://evernote.com/products/scannable

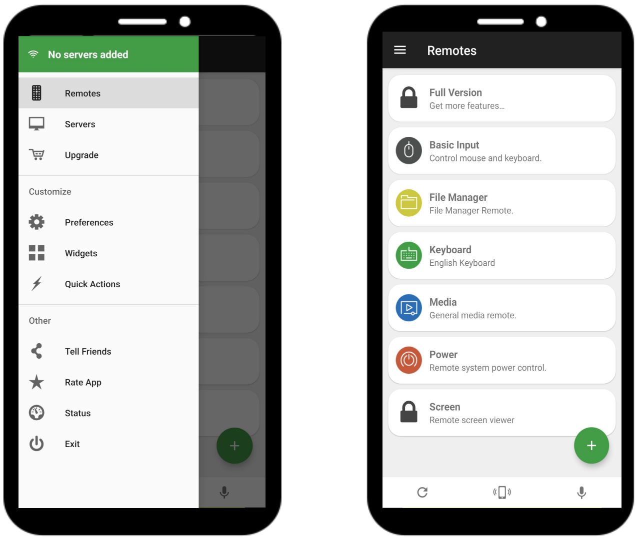

Figure 2.1: Set up screens of unified remote application for a wireless trackpad on Android

device and our computer. This app lets us control our computer's mouse, keyboard, and media playback using our device's touchscreen and sensors. We can also access program-specific remotes for presentation control and other advanced features[20]. Another alternative could be KDE Connect[1], a multi-platform application that allows wireless communication and data transfer between devices on the same network. It is developed by KDE, a community of software developers that create free and open-source software for desktop and mobile platforms. KDE Connect can be installed on Linux distributions from their repositories or F-Droid and Android devices from the Google Play Store. Some Linux distributions include KDE Connect in their KDE Plasma desktop version. KDE Connect has also been adapted for the GNOME desktop environment as GSConnect, which can be downloaded from the Gnome Extension Store.

- *Remote computer terminal*: Another way to repurpose an old Android device is to turn it into a remote computer terminal. We can access our home or office computer from anywhere using our device's screen. We can use a third-party app called TeamViewer, a remote access and remote control

[1] https://kdeconnect.kde.org/

computer software to establish a secure connection between our device and computer. We can then view and control our computer's desktop, files, and applications from our device[21]. Other alternatives are RealVNC, remote desktop, moonlight, and other gaming clients such as Steam, which offers a two-step verification to users who want to protect their accounts[1].

- *Universal smart remote*: We can use our old Android device as a universal smart remote. Using our device, we can control various smart devices in our home, such as lights, thermostats, cameras, and locks. We can use open-source apps like Home Assistant which focuses on privacy and local control, and paid apps like Yeti Smart Home[22] to connect and manage all our compatible devices from one place. We can also turn it into a media remote for Kodi[2]. This software allows us to enjoy our home theatre with a remote control. We can use it on different devices, such as the RPi and Android, with their remotes. Kodi has a user interface that is designed for a large screen and a remote, so using a keyboard or a mouse is not recommended or supported by some skins. Another choice will be to make a remote control for Plex, a free streaming app for everyone[3].
- *Power scientific research*: Another way to utilize an old Android device is to let it power scientific research. We can install apps that use our device's idle computing power to contribute to various research projects. For example, we can use an app like DreamLab, as shown in Figure 2.2, to help researchers find treatments for cancer, COVID-19, and other diseases. We can also use an app like MyShake to help scientists detect earthquakes and provide early warnings[23] even though this method seems to be not energy efficient as the device's idle computing power is not free because it consumes electricity.
- *A data-based extension*: We can make our old phone a data-based extension for our current phone service[24]. If we use Google Fi (formerly known as Project Fi) for our current phone's wireless service, we can take advantage of a feature that lets us use the same number on multiple devices. We can add our old device to our plan for free and use it to make calls, send texts,

[1] https://steam.en.uptodown.com/android
[2] https://kodi.wiki/view/Remote_controls
[3] https://www.plex.tv/

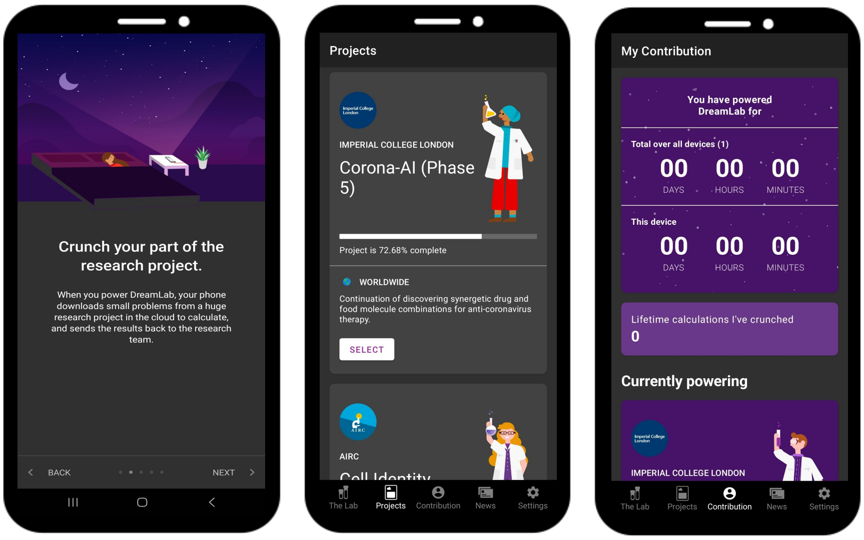

Figure 2.2: DreamLab application screens showing how to select a project to participate in.

and access data over Wi-Fi or cellular networks. This way, we can have a backup phone in case our main one runs out of battery or gets lost.

- *Personal testing ground*: If we would like to experiment with different apps, settings, launchers, themes, wallpapers, widgets, and other customizations on our Android device. In that case, we can use our old device as a sandbox for testing them out. This way, we will keep our primary device and all important data. We can also try out different versions of Android by installing custom ROMs or using emulators on our old devices [25].

C. Several ways to make use of our old Windows phone:

- Use it as a fully functional PC with the Continuum feature and a NexDock or a Microsoft display dock.
- Use it as a personal home cloud storage with the FTP server app and access our files from any device on the same WiFi network.
- Use it as a remote control, mouse, or keyboard for our PC with the PC Remote app and server.

- Use it as a fitness tracker with the Runtastic[1] or Caledos[2] runner app and track our physical activity and calories burned.

2.3 Distributed computing

Distributed computing is a field of computer science that studies the design, analysis, and implementation of multiple computers or devices that communicate and coordinate over a network. Distributed computing aims to solve complex problems that require high performance, scalability, reliability, and availability by leveraging the resources and capabilities of multiple devices or systems. Distributed computing also enables applications and services that are distributed, decentralized, or parallel, such as cloud computing, peer-to-peer systems, distributed databases, and distributed machine learning.

Distributed computing has many challenges and research topics, such as distributed algorithms, distributed systems architecture, operating systems, middleware, communication protocols, security, fault-tolerance, distributed synchronization, distributed resource management, and distributed testing and verification. Some of the well-known models and paradigms of distributed computing are:

- Cloud computing is a model of distributed computing that provides on-demand access to shared resources, such as computing power, networking, and storage, over the Internet[26]. Cloud computing can offer various services, such as infrastructure as a service (IaaS), software as a service (SaaS), platform as a service (PaaS), and function as a service (FaaS). Cloud computing can also be classified into different types, such as public cloud, private cloud, hybrid cloud, and multi-cloud. Some benefits of cloud computing are cost-efficiency, scalability, elasticity, and availability. Some of the challenges of cloud computing are security, privacy, interoperability, and latency. Some examples of cloud computing platforms are Amazon Web Services, Google Cloud Platform, Microsoft Azure, and IBM Cloud[26].

[1] https://www.runtastic.com/challenges/runtastic/ublight_23
[2] https://www.caledos.com/

- Peer-to-peer systems: Peer-to-peer systems are distributed systems composed of autonomous and self-organizing nodes that can act as clients and servers and communicate directly without relying on a central authority or intermediary. Peer-to-peer systems can offer various functionalities, such as file sharing, content distribution, collaborative computing, social networking, and distributed computing. Some advantages of peer-to-peer systems are resilience, scalability, anonymity, and decentralization. Some disadvantages of peer-to-peer systems are security, reliability, quality of service, and incentive compatibility. Some of the examples of peer-to-peer systems are BitTorrent, Gnutella, Skype, and Bitcoin[26] [27].

- Distributed machine learning: Distributed machine learning is a branch of machine learning that deals with the design and implementation of machine learning algorithms that can run on distributed systems, such as clusters, grids, clouds, or edge devices. Distributed machine learning can enable large-scale and complex machine learning tasks, such as data analysis, image processing, natural language processing, and deep learning, by exploiting distributed systems' parallelism, distribution, and heterogeneity. Some of the benefits of distributed machine learning are speed, scalability, robustness, and privacy. Some challenges of distributed machine learning are communication, synchronization, coordination, and consistency. Some of the examples of distributed machine learning frameworks are TensorFlow, PyTorch, Apache Spark, and Hadoop[26].

2.4 Micro-clouds and cloudlets

Micro-clouds and cloudlets are two emerging technologies that can enhance the performance and functionality of mobile and edge computing. They are both small-scale cloud data centers that are located closer to the end users, providing lower latency, higher security, and better privacy than the traditional cloud. However, they also have some design, implementation, and application differences. This section will compare and contrast micro-clouds and cloudlets and explore how they can benefit various use cases, such as distributed computing, augmented reality, wearable cognitive assistance, speech recognition, and edge analytics. We will also discuss some of the challenges and opportunities for developing and deploying micro-clouds and cloudlets in the future.

2.4.1 What are micro-clouds?

Micro-clouds are a new class of infrastructure that can be used for edge computing, as shown in Figure 2.3. Edge computing is a topology that describes geographically distributed computing, where data is processed closer to the source or the user rather than in a centralized cloud[28]. Another term used is Fog computing, a paradigm that extends cloud computing to the network's edge, where devices such as sensors, actuators, and smartphones can participate in data processing and service delivery. Fog computing aims to reduce latency, bandwidth, and energy consumption by bringing computation closer to the data sources and consumers. However, fog computing faces challenges such as resource heterogeneity, scalability, security, and reliability[29]. A term associated with fog computing is Gateways, which connect different networks and protocols, such as IoT devices and the cloud. Gateways can perform various functions, such as data aggregation, filtering, compression, encryption, and protocol translation[30]. Gateways can also act as smart gateways, hosting some fog services and applications and providing security and privacy mechanisms. Smart gateways can work together with micro-clouds to deliver the fog vision.

Micro-clouds prioritize decentralized environments' security, privacy, governance, and low latency over the scalability offered by public and private clouds. They mirror the APIs and foundational components of major cloud services but operate at the edge, comprising small clusters replicated across numerous locations [31]. Micro-clouds can help build a successful edge computing strategy with open-source technologies and solutions[32].

Small, cheap, and low-power computers have opened up new possibilities for applications such as smart home and entertainment systems. Some projects and companies have used these devices to build small clusters of computers for various purposes (for example, PiFace Rack1, a rack of Raspberry Pi (rPi) devices)[33]. These devices, coupled with advancements in virtualization and hypervisor technologies facilitating resource sharing within isolated environments, have paved the way for developing micro-cloud systems. Micro-clouds are small-scale computational infrastructures that can be easily deployed anywhere. They differ from the large data centers that power the cloud but can offer similar features, such as on-demand and pay-as-you-go access to resources. It is essential to note the difference between micro-clouds and other terms, such as

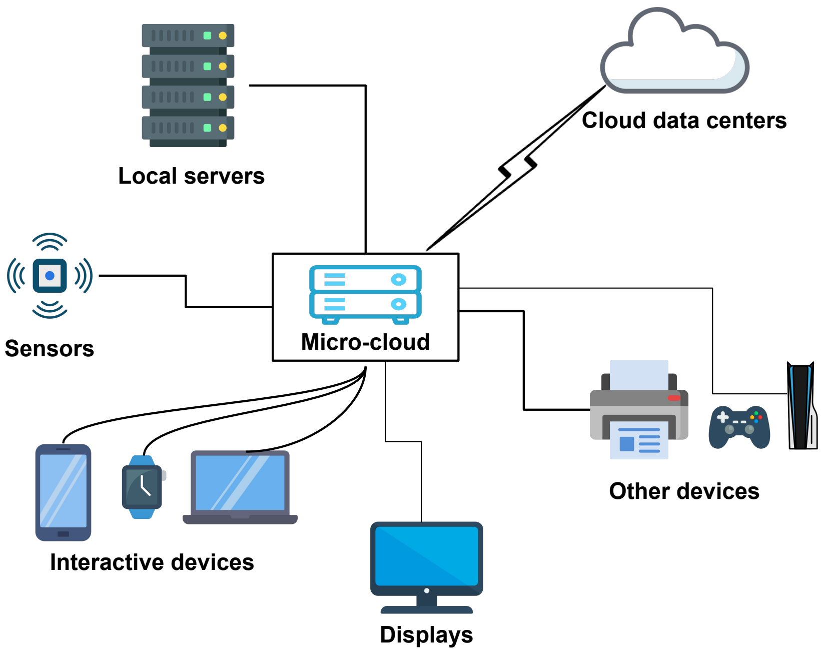

Figure 2.3: A micro-cloud interconnecting a swarm of edge devices.

mini-clouds and mini-data centers. The latter refers to modular server racks that are deployed indoors to provide high computing and storage capabilities.

An early example is Sun's data center in a shipping container [34]. More recent examples are much smaller but require a 19-inch server rack enclosure. Micro-clouds also consist of modular computers, which are more portable and independent of existing infrastructure (such as a server room with temperature and humidity control). Therefore, micro-clouds can be deployed outdoors and indoors, especially in harsh or unprepared environments.

One of the applications of micro-clouds is to support data science in underwater environments where traditional computing and communication technologies are limited. Underwater sensors can collect various types of data, such as temperature, salinity, pressure, sound, and images, but they face challenges in transmitting and processing them in real-time. To address this problem, this study [35] has proposed using low-cost micro-clouds as underwater fog computing platforms that provide local data analysis and storage capabilities. They have built a prototype of an underwater micro-cloud using off-the-shelf devices and tested it in different aquatic scenarios, such as river and sea waters. They have also evaluated the performance of different communication interfaces, such as Wi-Fi, Bluetooth, and radio frequency, and developed an accelerometer method to detect communication failures and switch between interfaces. Their work shows that underwater micro-clouds can enable more scalable and complex data science applications in aquatic areas.

Another area where micro-clouds can be useful is fog computing, which aims to leverage the computing power of edge devices to reduce the dependency on the cloud. Fog computing can benefit applications that require low latency, high bandwidth, or local data processing. However, fog computing faces challenges such as resource heterogeneity, security, and scalability[31]. Micro-clouds, small and portable clusters of single-board computers, can solve these challenges by providing isolated and cost-effective resource provisioning at the edge. This study [31] has explored the feasibility and readiness of micro-clouds for fog computing by using Raspberry Pi devices as the building blocks. They have demonstrated that micro-clouds can support various fog-related applications, especially in locations with limited or unreliable network connectivity.

2.4.2 What are cloudlets?

Cloudlet is a small-scale data center located at the edge of the Internet to provide computing resources to mobile applications with lower latency. It is a newer architectural element of cloud computing infrastructure of a 3-tier hierarchy: mobile device or IoT hardware, cloudlet, and cloud. The cloudlet concept is also known as mobile micro-cloud or edge cloud [36].

Cloudlets are a new concept that has been studied in various aspects. For example, this study [37] explored the challenges and applications of cloudlets from the point of view of a thin and thick client. [38] examined the potential benefits of cloudlets in wireless local area networks (WLANs). They also identified technical difficulties in deploying cloudlets in local area networks (LAN) commercially, such as scalability, mobility, and deployment cost [39]. Their work mainly focused on the energy efficiency of cloudlets and the role of different algorithms in reducing energy consumption. Another study [40] proposed a two-tier cloud in a cloudlet architecture using fiber-wireless (FiWi) networks. They also discussed the suitability of mobile edge computing for emerging applications, such as AR, VR, cognitive assistance, and cloud robotics, in the context of future 5G mobile networks moving toward cloudlet-based decentralization. Furthermore, this study [41] provided a brief review of computation offloading methods to address the resource limitations of mobile devices. It highlighted some recent developments in computation offloading in the industry, such as MAUI [42], Oddessa [43], CloneCloud [44], and COMET [45].

Cloudlets have many benefits, such as reducing communication latency and improving connectivity. Cloudlets use Wi-Fi connections and save the battery life of mobile devices by offloading heavy tasks to a cloudlet for processing. Cloudlets and an enterprise cloud, such as Google, can provide applications and services to users. Moreover, cloudlets do not need to access an enterprise cloud to synchronize with other cloudlets. However, when accessing a company cloud is required, cloudlets process most of the data and transmit less traffic to the cloud, resulting in low data storage requirements, low bandwidth, and lower communication latency, delays, and jitters [46]. Besides these main advantages, cloudlets also offer essential benefits in interactive gaming environments, user satisfaction, privacy, and security.

Micro-clouds and cloudlets have some similarities and some differences. They both aim to provide low-latency computing resources at the edge of the network, but they have different approaches and architectures. Some of the main points of comparison are:

- Micro-clouds are based on containerization technologies, while cloudlets are based on virtualization technologies.
- Micro-clouds use Kubernetes as the orchestration layer, while cloudlets use OpenStack as the management layer.
- Micro-clouds support various applications and services, while cloudlets focus on offloading computation from mobile devices.
- Micro-clouds are more flexible and scalable, while cloudlets are more stable and reliable.

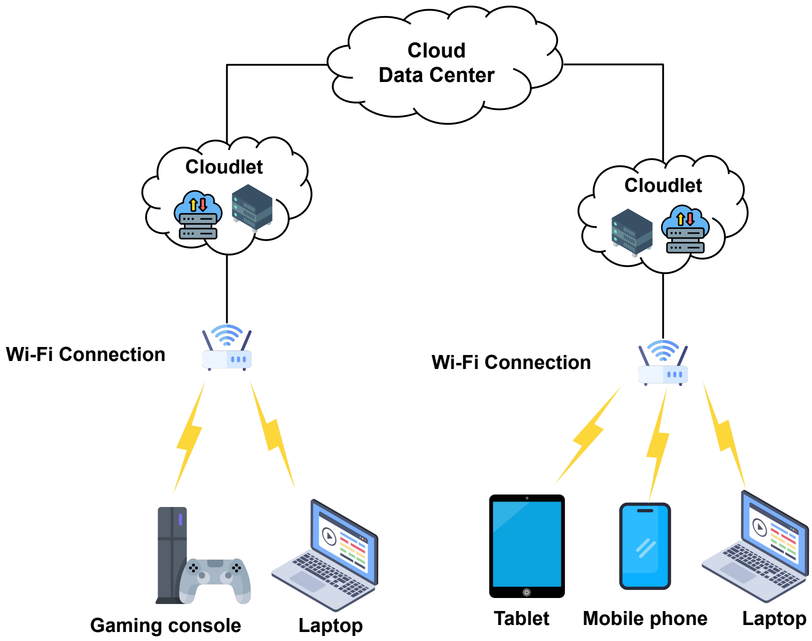

Figure 2.4: A simplified cloudlet architecture.

2.4.3 Survey of Research on Micro-Clouds and Cloudlets

A study has demonstrated that a smartphone rack can be used to create a cloud computing-like infrastructure[47]. This study introduces Misco, a framework that enables distributed computing on mobile devices using the MapReduce model. The study shows how Misco can run on Python-enabled devices like Nokia N95 smartphones and handle failures, concurrency, and communication issues. The study evaluates Misco's performance, overhead, and feasibility using different applications and failure rates. The study also discusses the challenges and opportunities of using mobile devices as data producers and consumers in a MapReduce framework. The study claims that Misco is the first system of its kind and proposes future directions for research.

Some studies have suggested using smart and IoT devices as mini data centers that can work together. The first paper proposes a resource management framework for Fog computing[48]. This paradigm brings cloud resources closer to the edge of the network, where IoT devices and sensors are located. The paper considers the challenges of dealing with heterogeneous devices with different types of data, services, and connectivity. The paper introduces the concept of relinquish probability, which is the probability that a device will stop using the resources it has requested. The paper uses this concept to estimate the right amount of resources for each device based on the service type, service price, and variance of the relinquish probability. The paper evaluates the framework using Java and CloudSim. It shows that it can improve the resource utilization and profit for the service providers and the Fog. The second paper explores the idea of using non-volatile memory (NVM) on mobile devices to cache cloud services and improve their latency and energy efficiency[49]. The paper proposes a pocket cloudlet architecture that uses individual and community access models to increase the cache hit rate. The paper demonstrates the concept with PocketSearch, a web search and advertisement pocket cloudlet that can serve most queries locally without using the 3G network. The paper evaluates PocketSearch with real mobile queries from Bing and shows that it can achieve significant speedup and energy saving. The paper also gives guidelines and insights for future pocket cloudlet design and implementation.

Moreover, several studies have also explored how collaborative processing and federated learning between devices can be used to create dynamic and elastic computing infrastructures on the edge. The first paper introduces the femtocloud system[50], a framework that allows a group of co-located mobile devices to form a cloud service at the edge of the network. The paper argues that this system can utilize mobile devices' underused computing capabilities and improve cloud services' latency and energy efficiency. The paper describes the design and architecture of the femtocloud system, which includes a task scheduling problem that considers the connectivity and execution time of each device. The paper evaluates the femtocloud system using a prototype and simulations. Furthermore, it compares it with an oracle with perfect system knowledge. The paper shows that the femtocloud system can achieve high compute throughput and scalability and is close to the oracle's performance. On the other hand, the second paper presents a novel approach to applying deep learning for IoT devices in an edge computing environment[51]. The paper argues that deep learning can extract useful information from raw sensor data and improve IoT devices' network performance and privacy. The paper also proposes a new offloading strategy to optimize the execution of deep learning tasks on edge nodes with limited processing capability. The paper evaluates the approach using ten different CNN models as deep learning networks and compares it with other optimization solutions. The paper shows that the approach can increase the number of tasks deployed on edge servers with guaranteed QoS (Quality of Service) requirements.

In addition to the studies above, another study has experimented with micro-clouds on flying drones at the edge[52]. The study proposes Cloudrone, a novel idea of creating a micro cloud infrastructure in the sky using low-cost drones, single-board computers, and lightweight virtualization technologies. The study argues that Cloudrone can be deployed on demand and provide various services to users. The paper presents an initial design of Cloudrone and evaluates its scalability using simulations.

Finally, another study presents a novel approach to enabling underwater data science applications using submersible fog computing[53]. The authors use low-cost micro-clouds, off-the-shelf devices placed in waterproof containers, to augment the computational capability of underwater sensors. They evaluate the performance of their approach in different underwater settings, such as river and sea waters, and show how water motion and depth affect communication and computation. They also propose a method to use accelerometer-based sensing to determine the best time and interface for communication. Furthermore, this study demonstrates the feasibility and benefits of using submersible microclouds for various underwater data science scenarios.

2.5 Summary

This Chapter reviews the related work on reusing mobile phones, micro-clouds and cloudlets, and distributed computing. Reusing mobile phones is repurposing and recycling old or unused ones for various purposes, such as security cameras, remote computer terminals, a data-based extension, or cloudlet nodes. Micro-clouds and cloudlets are small and portable data centers located at the edge of the Internet to provide low-latency and high-performance computing resources to mobile devices and applications. In contrast, Distributed computing is the field and implementation of systems that consist of multiple computers or devices that communicate and coordinate over a network to solve complex problems, using models and paradigms such as cloud computing, peer-to-peer systems, and distributed machine learning.

3

Methodology

Next, we aim to explain the steps of our proposed lightweight cloudlet for providing fog computing services for different applications. Our methodology comprises different phases: design, implementation, and evaluation. In the design phase, we define the requirements and specifications of our cloudlet, such as the hardware components, the software stack, the communication interfaces, and the power supply. We also describe the case's design and 3D printing process. In the implementation phase, we assemble and configure our cloudlet prototype using off-the-shelf devices and open-source software. We also develop the applications and services we want to run on our cloudlet, such as distributed computing, web applications, and web servers. In the evaluation phase, we test the performance of our cloudlet in different scenarios. We measure our cloudlet's performance, reliability, and energy efficiency, as well as the quality of service.

3.1 Feasibility and opportunities

Regarding software, we tested the feasibility of installing PostmarketOS Linux-based mobile OS on Samsung Galaxy S3 and Nexus 5 devices. We tried different user interfaces provided by PostmarketOS. We found that the Samsung Galaxy S3 was buggy and did not support the USB feature over Ethernet. USB over Ethernet is a system to share USB-based devices over Ethernet, Wi-Fi, or the Internet, allowing access to devices over a network (allowing SSH connection via USB). Without USB over Ethernet, it was difficult for us to do any debugging or troubleshooting related to the phone as we could not connect to it. Furthermore, the screen of the Samsung Galaxy S3 faced some severe issues in terms of freezing. The phone could freeze for multiple hours; the only solution was to restart it repeatedly. On the other hand, the Nexus 5 supported USB over Ethernet and did not experience the screen freezing issue.

Regarding hardware, we collected several obsolete phones, as shown in Table 3.1. Note that apart from the Samsung and the Nexus, all other devices were not working. This was good for us in terms of disassembling the phones. However, we disassembled at least one Nexus 5 phone in working conditions for experimental purposes. Among all our phones, the Nexus 5 was the best choice hardware-wise because the phone and its components were more accessible to teardown. Furthermore, its battery configuration allowed us to remove the battery circuits, which was one of our research questions. Additionally, we also needed to know the weight of each critical component of the phone, such as the motherboard, display, and battery. Learning the weights of these components was essential because our goal was to find out if we could use the motherboard alone, without other parts, and create a lightweight rack of motherboards. However, we realized that the motherboard could not work without connecting to the different parts of the phone, such as the display, the power button, and the USB port. This was because the motherboard relies on these parts for booting, controlling, and communicating with the phone. Therefore, we had to find a way to bypass or emulate these parts to make the motherboard work independently, which was unsuccessful.

Despite these challenges, we also saw some opportunities for using PostmarketOS and Nexus 5 for our project. One option was that PostmarketOS offered a variety of user interfaces and applications that could suit different users' needs and preferences. Not only the graphical interface but also provides a minimal environment with no UIs (console). Another opportunity was that the Nexus 5 had relatively powerful and modern hardware compared to its next model, the Nexus 6P. The Nexus 5 has a quad-core Snapdragon 800 processor, 2 GB of RAM, and 16 or 32 GB of internal storage. In contrast, the Nexus 6P, on the other hand, had an octa-core Snapdragon 810 processor, 3 GB of RAM, and 32, 64, or 128 GB of internal storage. Although the Nexus 6P had more cores, RAM, and storage than the Nexus 5, it also had more overheating, throttling, and battery degradation issues[54]. Therefore, we decided to use the Nexus 5 for our experiments, as it was more stable, reliable, and compatible with PostmarketOS.

Devices	Model	Total weight(g)	Without battery (g)	Motherboard (g)	Display (g)
Huawei nexus 6p	HB416683ECW	179	102	24	77
Nexus 5	LG-D821	130	80	13	50
Oneplus X	E1003	138	70	13	56
Samsung Galaxy S3	GT-I9505	133	87	12	53
Samsung Galaxy S1	GT-S6102	119	77	17	33
Sony Z1	D5803	119	69	21	47

Table 3.1: Weights of different device components.

3.2 Hardware requirements

In order to achieve the proposed cloudlet, we collected six unused Nexus 5 (LG-hammerhead) phones from the distribution system lab of the University of Tartu. Four phones were in perfect working condition, and the remaining two had their battery not working. The following section provides the hardware details of the components used to build the cloudlet.

3.2.1 Nexus 5 (LG-hammerhead)

The Nexus 5 is a smartphone launched in 2013 by Google and LG[1]. It has a 4.95-inch display with a resolution of 1080x1920 pixels, which gives it a pixel density of 445 ppi. The Nexus 5 is powered by a Qualcomm Snapdragon 800 processor (MSM8974) with four cores clocked at 2.3 GHz and an Adreno 330 GPU. The Nexus 5 has 2 GB of RAM and 16 GB or 32 GB of internal storage but no microSD card slot. The Nexus 5 has an 8 MP rear camera with optical image stabilization, LED flash, and 1080p video recording. The front camera is 1.3 MP and can record 720p video. The Nexus 5 has a 2300 mAh battery that supports fast and wireless charging. The Nexus 5 runs on Android 4.4 KitKat, the first Android version to support 64-bit architecture. It also has a fingerprint scanner, NFC (near-field communication), and dual microphones. The Nexus 5 was one of the flagship devices of the Nexus series, which was designed by Google and offered a pure Android experience.

[1] https://en.wikipedia.org/wiki/Nexus_5

3.2.2 DFRobot DRF08x Fast Charge Buck Modules

In order to power the phones, we used the DFRobot DRF08x Fast Charge Buck Modules[1], which can charge mobile phones quickly and efficiently. We used this module to power four mobile phones simultaneously instead of having four power outlets. These modules have a particular function that can recognize most mobile phones' voltage and current requirements. They come in two types: 2-way and 4-way, which can charge two or four devices simultaneously. They are also compatible with Raspberry Pi 4B and Jetson Nano. The DRF08x fast charge buck modules can work with different input voltages from 6V to 32V and output different voltages from 3V to 12V (default 5V). The output voltage can be changed by using a fast-charging decoy device. These modules support multiple fast-charging protocols, which can trigger the maximum current for fast charging and save time. The DRF08x fast charge buck modules have a stable output that does not change over time. These modules suit mobile phone fast charging, car charging retrofit, solar chargers, and DIY power sources.

3.2.3 GL-SF1200 AC1200 Wireless Gigabit Router

In order to facilitate the communication between the devices forming the cloudlet, we chose to use GL-SF1200 AC1200 Wireless Gigabit Router[2] to provide Wi-Fi connectivity for our cloudlet. This router is a cost-effective and powerful device that offers dual-band Wi-Fi and four external high-gain antennas, which are ideal for families and businesses that need fast and reliable Wi-Fi. The router also has security features that protect our internet privacy, such as OpenVPN and WireGuard, which are encryption and authentication protocols that allow us to create a private and secure connection to the internet. The router also has a web Admin Panel, a user-friendly interface that lets us check the status and manage the router's settings. We use a VPN for our cloudlet to ensure the security and privacy of our data and devices. A VPN can prevent hackers, internet service providers, and other third parties from snooping on our internet activity and stealing our data. A VPN can also help us access our files securely on a remote site as if at home. By using a VPN, we can create a private and encrypted tunnel between our cloudlet and the internet, improving the performance and reliability of our cloudlet services.

[1] https://www.mouser.ee/new/dfrobot/dfrobot-drf08x-fast-charge-buck-modules/
[2] https://www.gl-inet.com/products/gl-sf1200/

3.3 Software requirements

To run the proposed cloudlet, we need to install and configure the software components that enable the functionality and communication of the cloudlet nodes. The following section describes the software details of the operating system, the network protocol, and the applications and services we use to build the cloudlet.

3.3.1 Android Debug Bridge (ADB)

Android Debug Bridge is an incredibly flexible command-line tool designed to enable seamless communication with a device. By utilizing the adb command, we can execute various device actions, including app installation and debugging. it grants access to a Unix shell, allowing us to execute diverse commands on our device. This client-server program comprises three essential components[1]:

How to set up ADB

1. Launch the **Settings** application on your phone.
    - Depending on the OEM skin, the **About phone** page may be called something else or buried somewhere else in the **Settings** app on your device.
2. Then tap the **Build number** option seven times to enable Developer Mode. You will see a toast message when it is done.
3. Now go back to the main Settings screen, and you should see a new **Developer options** menu you can access. On Google Pixel phones and some other devices, you might need to navigate to **Settings** > **System** to find the **Developer options** menu.
4. Go in there and enable the **USB debugging** option.
5. You are partially done with the phone setup process. Next up, you will need to scroll below and follow the rest of the instructions for your particular operating system.

[1]https://developer.android.com/tools/adb

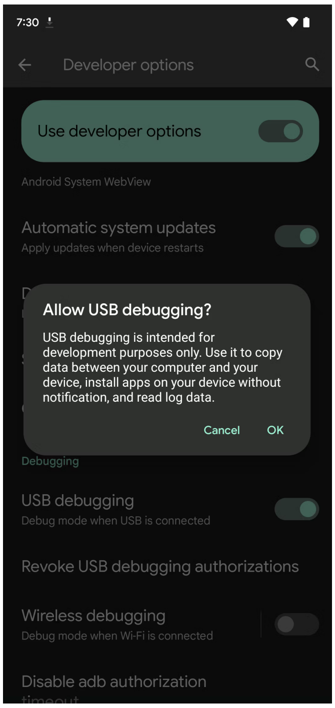

Figure 3.1: Allow Android debugging mode screen.

- The client, i.e PC/Mac/Chromebook, you have connected your Android device. We are sending commands to our device from the computer through the USB cable or wirelessly.
- A daemon (known as "adbd") that runs commands on a device. The daemon runs as a background process on each device.
- A server that manages communication between the client and the daemon. The server runs as a background process on the computer.

3.3.2 pmbootstrap

pmbootstrap is the central command-line application for postmarketOS development. It allows building packages to create installation images and flash them to your phone. If you only want to install postmarketOS, read the installation article first since you might not need to install pmbootstrap, depending on the method[1].

[1] https://wiki.postmarketos.org/wiki/Pmbootstrap

3.3.3 PostmarketOS

postmarketOS is an open source Linux-based operating system[1] that aims to provide a sustainable and secure alternative to Android for mobile devices. postmarketOS is different from Android in several ways, such as:

- It avoids Android's complex build system, which requires many resources. Instead, it uses a modular and lightweight approach, where the OS is divided into small packages that can be installed on any device with the same CPU architecture. Therefore, making the development faster and easier allows for more customization and flexibility.
- It is based on Alpine Linux, a minimal and secure Linux distribution that is less than 10 MB in size. Alpine Linux provides a simple and efficient package manager. postmarketOS uses Alpine Linux as its base system and installs it in multiple chroots to cross-compile packages, build and flash postmarketOS, run it in a Virtual Machine (VM), or port new hardware.
- It supports multiple user interfaces from independent projects, such as Plasma Mobile, Phosh, and Sxmo. These user interfaces offer different user features and experiences and can be easily switched or installed on postmarketOS. postmarketOS also supports various applications and services compatible with Linux, such as data analysis, image processing, and machine learning.
- It is designed to be up-to-date for all devices at once. Unlike Android, which often suffers from fragmentation and outdated versions, postmarketOS can provide regular updates and security patches for all devices, regardless of their manufacturer or model. postmarketOS also aims to extend the lifespan of old or unused devices by reusing them as cloudlet nodes, security cameras, gaming systems, or video chat devices.

3.4 Experiments

We conducted our experiments in two parts: the first part involved preparing the Nexus 5 devices for the cloudlet, and the second part involved assembling and testing the

[1] https://wiki.postmarketos.org/wiki/Main_Page

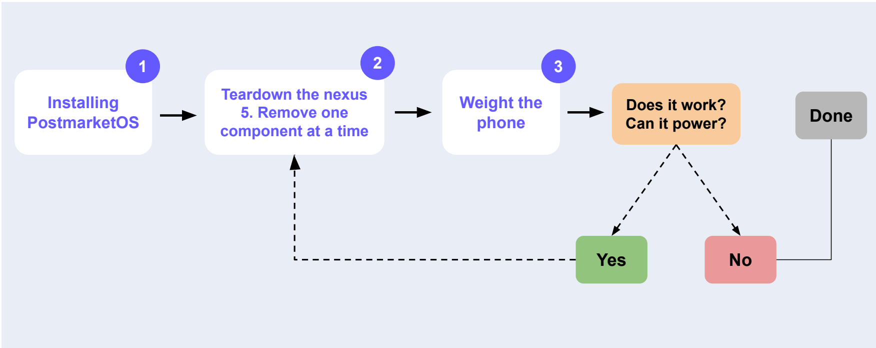

Figure 3.2: This figure provides a high-level overview of the steps of our experiments.

cloudlet. Figure 3.2 illustrates the steps of the first part, which included installing a Linux-based operating system to gain complete control (root access) and disassembling the devices carefully by removing one component at a time and ensuring they still function correctly. The second part involved connecting the devices to a router and a power source and configuring the cloudlet settings and services. The purpose of these experiments was to answer the research questions that we posed in the introduction.

3.4.1 PostmarketOS installation

To address the research question of how feasible it is to install Linux-based operating systems on the Nexus 5, we describe the steps of creating and flashing an image for the device using PostmarketOS.

Fastboot mode

First, we need to ensure that the USB debugging in Android is enabled. Then with the device powered off, hold the **Volume Down and Power** buttons simultaneously for a few seconds until the Fastboot Mode screen appears.

OEM unlock

Get the device into the Fastboot Mode, plug in the USB cable, then type
See the following command :

```
$ sudo fastboot oem unlock
```

When prompted on the device, use **Volume-Up/Down** buttons to highlight the choice, then the Power button to select it. The Fastboot Mode screen will appear again, and a message regarding flashing will appear briefly.

Install pmbootstrap on Linux

pmbootstrap offers the easiest way to install postmarketOS. On Linux, it can be installed as shown. Begin by installing Python 3.6 or above.
On Debian/Ubuntu:

```
$ sudo apt install python3 python3-pip git
```

On Rhel/CentOS/Rocky Linux 8/Alma Linux 8:

```
$ sudo yum install python3 python3-pip git
```

Now we can use PIP to install pmbootstrap:

```
$ pip3 install --user pmbootstrap
$ source ~/.profile
```

You can upgrade it with the command:

```
$ pip3 install --user --upgrade pmbootstrap
```

Alternatively, you can install the latest development version as shown:

```
$ git clone https://gitlab.com/postmarketOS/pmbootstrap.git
$ cd pmbootstrap && sudo python3 setup.py install
```

Flashing

Make sure that pmbootstrap is installed in your distribution.
Build the image :

```
$ pmbootstrap init
```

Press enter to select and **Work path**, then select the target device (in our case, Nexus 5):

```
$ lg-hammerhead
```

```
$ Enable this package? (y/n) [y]: y
$ Username [user]: <username>
```

Select which interface to install:

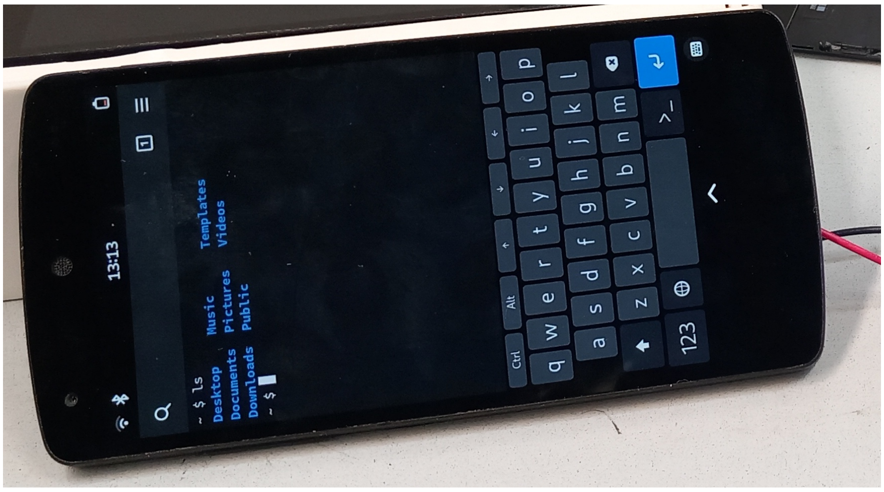

Figure 3.3: Picture of the Nexus 5 phone's terminal screen after postmarketOS installation.

```
$ User interface [weston]: plasma-mobile
$ Change them? (y/n) [n]: <Press Enter to select default>
$ Extra packages [none]: vim
$ Your host timezone: <default time zone will appear>
$ Use this timezone instead of GMT? (y/n) [y]:
$ Device hostname (short form, e.g. 'foo') [lg-hammerhead]: hammerhead
$ Would you like to copy your SSH public keys to the device? (y/n) [n]: y
$ Done!
```

Let's install all the selected dependencies:

```
$ pmbootstrap install --no-fde
$ New password:
$ Retype new password:
$ passwd: password updated successfully
$ Done
```

When the installation is done, turn off the phone, then press the Up Volume and Power buttons until the boot screen appears.

```
$ pmbootstrap flasher flash_rootfs --partition=userdata
$ pmbootstrap flasher flash_kernel
$ fastboot reboot
```

Start the SSH Daemon

If you are running postmarketOS from a pre-built image, the SSH daemon is disabled. Open the terminal on your phone and type the following to start it. Your sudo password is your PIN (147147 in pre-built images):

```
$ sudo service sshd start
```

Enable the service on every boot:

```
$ sudo rc-update add sshd
```

Find the IP of your phone

When connecting your phone via USB Network, then the IP of the phone is 172.16.42.1

First time login

Make sure your phone is on the same Wi-Fi network as your PC, or connect your phone via USB to your PC. Then type:

```
$ ssh username@172.16.42.1
```

in pre-built images, username defaults to **user** and your user's password or PIN is **147147**. Then you can adjust the IP as necessary.

3.4.2 Nexus 5 Teardown

To investigate the research question of how we can minimize the weight and size of mobile phones by eliminating redundant components and preserving their functionality as cloudlet nodes. We followed instructions provided by an educational platform[1] for repairing parts of everyday things such as mobile phones, electronics, appliances, and PCs for tearing down the Nexus 5. We started with removing the battery, which is usually the first component that needs to be replaced due to its limited lifespan and usage.

Step 1: Bypassing the battery

The Nexus 5 is connected to the battery using the leads of the BL-T9 battery of the Nexus 5, as shown in Figure 3.4.

[1]https://www.ifixit.com/

Figure 3.4: Google Nexus 5 battery BL-t9 2300 mAh.

We removed the outer plastic layer and examined the small circuit board between the battery terminals and the phone. The phone connector had four metal contacts. The two bigger ones probably carried power and ground. In comparison, the two smaller ones might be for data and clock signals for a communication protocol like I2C (Inter-Integrated Circuit), a bus interface connection protocol incorporated into devices for serial communication. However, we noticed that the bottom middle contact was electrically connected to the large right contact, so there were only three usable pins and one for communication.

Some printed pieces of information were on this side of the circuit board, as shown in the left Figure 3.5. BL-T9 is the model of this battery package. *UL94V-O* is likely the flammability rating of the device by Underwriters Laboratories[1]. The rest of the information, like *NXCT 50 31*, is unclear; maybe it relates to the PCB design revisions. The black ink *F9 DA* is probably a batch number or a similar identifier for manufacturing.

On the other side of the circuit board, as shown in right Figure 3.5, there are only three solder points on the phone connector, which matches what we noticed earlier. To the left of it, there is a label *LI176AH*. We wonder if it means "Lithium Ion" battery

[1] https://en.wikipedia.org/wiki/UL_94

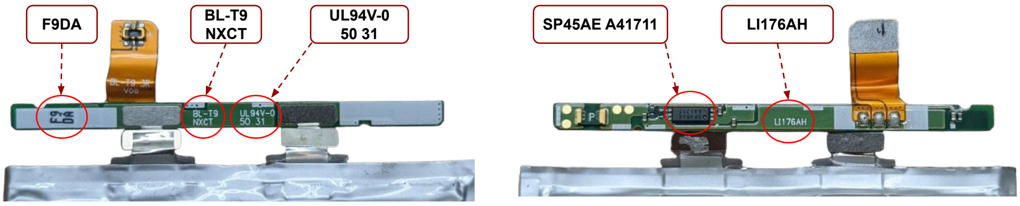

Figure 3.5: Google Nexus 5 battery circuit top and back.

with a specific Amp-Hour capacity, but 176 does not match the 2.2AH on the outside label, so it must have a different meaning.

Further to the left, we see a small chip with the mark *SP45AE A41711*, but we could not find any information about it online. Next to it, there are some test points and something we did not recognize with a *P* on it. It resembles a tiny circuit board with visible traces attached to the larger circuit board. We measured the resistance between the contacts on either side of the *P* and got 0.2ohm. We guess it is a shunt resistor for measuring current or a fuse, which could also be considered a current sensor.

Running on external DC power

In the first experiment, we attempted to run the Nexus phone on USB power without the battery, but it failed. The phone showed the low battery icon and then turned off. Therefore, we decided to modify the device to run on external DC power. We assumed that the device would not need a battery to buffer the fluctuations in power consumption, but we were not sure. As shown in Figure 3.6, we connected the ground wire to the negative terminal of the battery circuit and to the negative terminal of the controller.

We knew that the phone required 3.8V to power on, so we connected the positive and negative wires of the battery circuit to the output of the LM2596 DC-DC power supply, as shown in Figure 3.7. To test it, we used a 9V battery as the input of the LM2596, and the phone turned on right away.

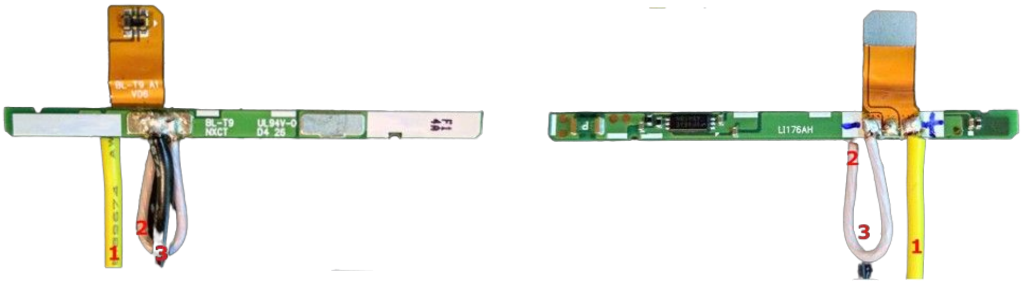

Figure 3.6: This figure shows the wires' connection to the battery circuit. 1 (+) and 2,3 (-).

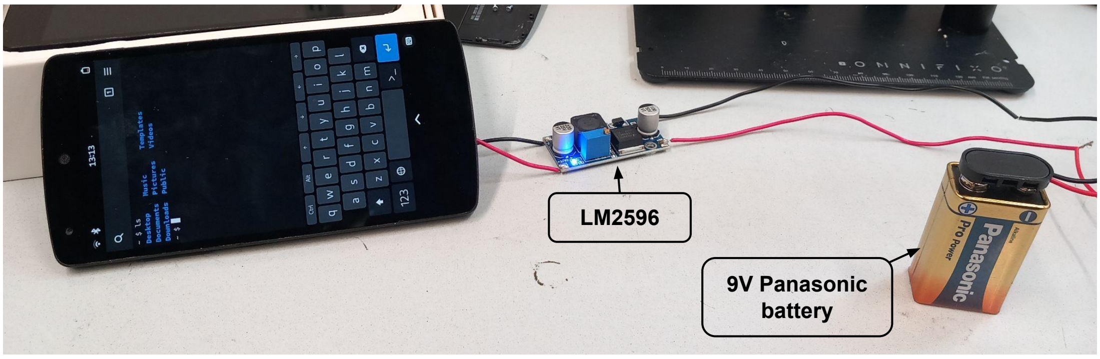

Figure 3.7: This figure shows the setup of the phone powered using a 9V battery connected to an LM2596 DC-DC.

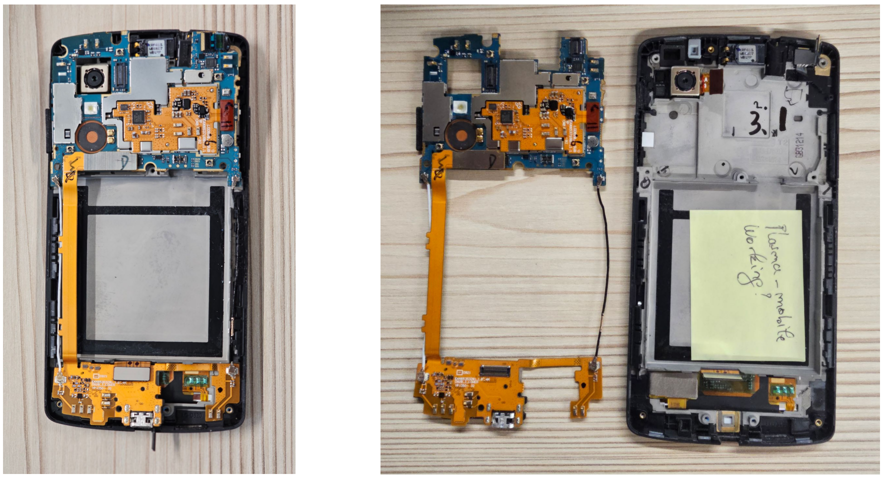

Figure 3.8: This figure shows the motherboard and daughterboard disconnected from the rest of the phone and reconnected to form a single component.

Step 2: Removing components

The second experiment involved removing unnecessary components from the phone. We removed one component at a time and checked if the phone still worked. The first component we removed was the speaker, which was easy to take out as it was only held by a few screws and had no cables attached. After removing the speaker, we powered the phone, and it turned on and functioned normally.

The next component we tried to remove was the daughterboard, as shown in Figure 3.9 as number 7, which was more complicated to remove as it had many components connected to it, such as the microphone, the RGB indicator LED, the micro-USB port, the speaker spring contacts, and the antenna spring contacts. Moreover, we discovered that the daughterboard was directly connected to the motherboard, so removing it would make the rest of the phone inoperable. We detached the daughterboard and the motherboard and reconnected them, as shown in Figure 3.8. We plugged the battery circuit into the remaining parts. We tested them, but the phone (motherboard and daughterboard) did not turn on, and we could not restore it.

3.4.3 3D printed encasing

3D printing is revolutionizing industries ranging from wearable vendors to sensor automation technologies [55, 56]. Besides this, 3D printing has also found its way into

Figure 3.9: Nexus 5 phone components disassembled with each number representing: 1. 8MP rear-facing camera, 2. Earpiece speaker, 3. Headphone jack, 4. 1.3MP front-facing camera, 5. Back case, 6. Motherboard, 7. Daughterboard, 8. LG D821 Nexus 5 Flex Platine PCB Assembly Sub, 9. Speaker, 10. Display, 11. Motherboard cover, 12. Battery.

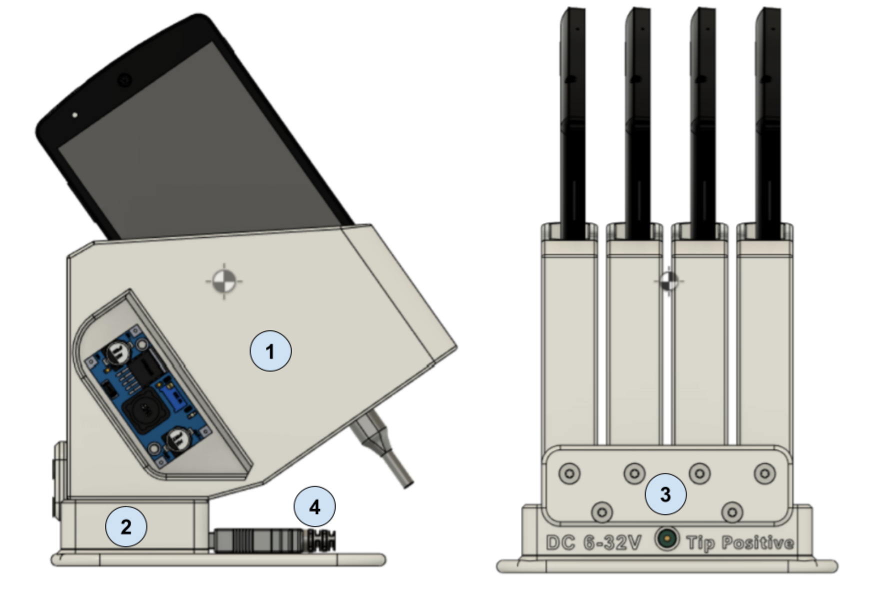

Figure 3.10: Overview of our 3D rack design encasing.(1) Cartridge, (2) Rack base, (3) Stabilizer bracket, (4) USB cable.

open hardware and DIY (Do-It-Yourself) communities, providing a large foundation of off-the-shelf model-based designs and a variety of different tools to build complex hardware solutions. 3D printing is mainly used to create components or parts that can be assembled into large physical objects. For instance, hand and machine tools, gadgets, or simply decorative and artistic objects [57]. Such are typically made with complex arrangements of vertices, shapes, and geometries without requiring moderate or high technical skills in making them.

To build our 3D model, we used Autodesk Fusion 360 (education version v.2.0.16985), a software that allows us to create and modify the 3D model of the case and simulate its functionality and performance. We designed our model to contain three most important parts: the phone, the DFRobot DRF08x Fast Charge Buck Modules, and the LM2596 DC-DC converter. The design can be seen in Figure 3.10. We designed our rack to allow both options, using an external power source or a battery-powered USB hub. This way, we can choose the most suitable and convenient power supply for our cloudlet, depending on the availability and cost of electricity. We also made our rack modular to add more components if required, such as more phones, fans, sensors, or LEDs. This gives our cloudlet more flexibility and scalability, as we can customize it according to our needs and preferences. All related files, such as the STL, STEP, and F3D files, can be found in the repository mentioned in the Appendix of this work. And we printed our model using a Prusa i3 model as a 3D printer at the University of Tartu.

3.5 Evaluation metrics of the Cloudlet performance

We used several evaluation metrics to measure the performance of each phone individually and collectively as a Cloudlet. We used the High-Performance Computing Challenge (HPCC) benchmarks, a set of tests designed to evaluate the performance of computer systems, particularly in the context of high-performance computing. These benchmarks cover various aspects of system performance, including computation, memory bandwidth, and communication. Here is an overview of four specific HPCC benchmarks:

1. **HPL (High-Performance LINPACK)**: This benchmark measures the floating-point rate of execution of solving a dense system of linear equations using the High-Performance Linpack (HPL) library. The performance metric is typically reported in GFLOPS (Giga Floating-point Operations Per Second). The formula for HPL is:
$$HPL = \frac{2}{3}n^3 + 2n^2$$
where n is the problem size in terms of the number of unknowns.

2. **Stream**: This benchmark measures the sustainable memory bandwidth of a system by performing four simple vector kernels: Copy (simple data copy), Scale (multiplication of each element by a scalar), Add (addition of corresponding elements of two arrays), and Triad (a combination of scale and add). The performance metric is reported in GB/s (GigaBytes per second); the formula for Stream is:
$$Stream = \frac{bytes}{time}$$
where $bytes$ is the number of bytes transferred in the kernel, and $time$ is the execution time.

3. **Random Access**: This benchmark measures the rate of integer random updates of memory (GigaUpdates per Second-GUPS) by performing a table update using a random address stream. It performs random reads and writes to a large array of memory locations; the formula for Random Access is as follows:

$$RandomAccess = \frac{updates}{time}$$

where *updates* is the number of table elements updated, and *time* is the execution time.

4. **FFT (Fast Fourier Transform)**: This benchmark measures the floating-point rate of execution of a double-precision complex one-dimensional Discrete Fourier Transform (DFT) using the Cooley-Tukey algorithm[1]. The performance is typically reported in FLOPS (Floating-point Operations Per Second). The formula for FFT is:

$$FFT = \frac{5n \log_2 n}{time}$$

where n is the transform size and *time* is the execution time.

3.6 Summary

This Chapter describes how we designed, implemented, and evaluated a lightweight cloudlet using off-the-shelf devices and open-source software. The section explains the requirements and specifications of the cloudlet, such as the hardware components, the software stack, the communication interfaces, and the power supply. The section also describes different metrics used to evaluate our cloudlets' performance, such as HPL, Stream, FFT, and Random Access.

[1]https://en.wikipedia.org/wiki/Cooley-Tukey_FFT_algorithm

4

Results

In this Chapter, we present and discuss the results of our evaluation of the proposed lightweight cloudlet. We report the results of different experiments that we conducted to assess the feasibility and effectiveness of our cloudlet in various scenarios, trying to answer the research questions proposed in Chapter 1. We compare our cloudlet with other existing solutions and discuss the advantages and limitations of our approach. We also analyze the impact of different factors on the cloudlet's performance and energy consumption, such as the number of devices, the type of applications, and the network conditions. We use different metrics to measure the quality of service our cloudlet provides

4.1 RQ1: Reducing the weight and size of mobile phones

One of our research objectives is to answer the question: How can we reduce the weight and size of mobile phones by removing unnecessary components and still maintaining their functionality as cloudlet nodes? To address this question, we conducted an experiment with Nexus 5 phones, which we used as the building blocks of our cloudlet. We disassembled the phones and tried to remove the components we deemed unnecessary for the cloudlet operation, such as the battery, the speaker, the camera, and the screen. We then tested the functionality of the phones by connecting them to a power source and a network switch.

We discovered that the only component we could remove from the phone and still make it work as a cloudlet node was the battery. Removing the battery reduced the phone's weight by about 20 grams and the thickness by about 2 millimeters. However, when we tried to remove the next component, which was the speaker, the phone stopped working. We could not find any apparent reason for this, as the speaker seemed independent from the main board and the network interface. We hypothesized that the phone had some built-in mechanism to detect the speaker's presence and prevent it from booting if missing.

We also encountered another problem when we tried reassembling the first phone we disassembled. The phone did not work even after we put back all the components that we removed. We suspected we might have damaged some of the delicate connectors or wires during disassembly. We realized that we did not follow all the precautions of disassembling the phone, such as using the proper tools, wearing anti-static gloves, and keeping track of the screws and cables. Therefore, we also provided some guidelines and precautions to follow when disassembling a mobile phone, which can be found in the Appendix of this work.

Another reason we could not reduce the weight and size of the phone by much was the evolution of technology. Modern phones are designed to be compact and integrated, with minimal space between the components. This makes them difficult to disassemble and repair, as the components are often soldered or glued together. Moreover, some components, such as the screen and the camera, are essential for the phone's functionality, as they provide the user interface and the input/output capabilities. Removing them would make the phone unusable as a standalone device and would require additional hardware and software to replace them.

4.2 RQ2: Power the Nexus 5 phone without its battery

To answer this research question, we conducted an experiment with a Nexus 5 phone, which we modified to run without its battery using alternative energy sources. We faced several challenges and benefits in this process, described below.

- The battery is one of the most dangerous components of the phone because it contains toxic and heavy metals that can leak, explode, or catch fire if damaged or overheated. The battery also adds weight and size to the phone, which reduces its portability and deployability as a cloudlet node. Moreover, the battery is a

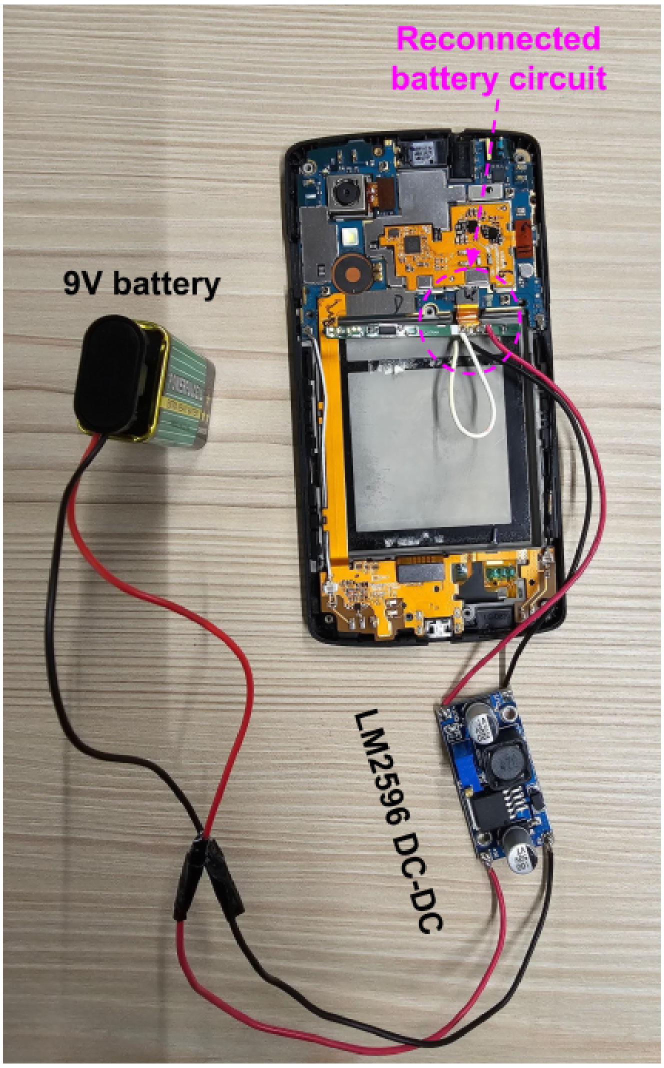

Figure 4.1: This figure illustrates the battery circuit modification for the Nexus 5, which enables the phone to run on a 9V battery.

non-renewable form of energy that relies on fossil fuels and contributes to environmental pollution. We carefully disassembled the phone and removed the battery and the circuit board, connecting it to the other components. We studied the different pins and marks on the circuit board and discovered how to supply power to the phone using an external DC power source (LM2596), as shown in Figure 4.1. Therefore, it allows us to power the phone with different energy sources, such as solar panels, wind turbines, or thermoelectric generators, depending on the availability and suitability of each source for the specific scenario. We also used a voltage regulator to adjust the output voltage of the power source to match the input voltage of the phone.

- Using alternative energy sources to power the phone is beneficial for several reasons. First, it can extend the lifetime and availability of the phone as a cloudlet node since it does not depend on the limited capacity and durability of the battery. Second, it can enable the phone to operate in remote or disaster areas where power supply is scarce or unreliable by harvesting energy from the environment. Third, it can reduce the dependency on batteries, which are costly, bulky, and harmful to the environment.

- Using alternative energy sources to power the phone is also sustainable because it reduces the consumption of fossil fuels and the emission of greenhouse gases. Alternative energy sources are renewable, meaning they can be replenished naturally and do not deplete over time. They are also cleaner, meaning that they produce less pollution and waste than fossil fuels. By using alternative energy sources, we can conserve natural resources and protect the environment.

- By bypassing the battery, we have some advantages regarding the performance and design of the phone as a cloudlet node. First, we can reduce the phone's weight and size, making it more portable and easily deployable in different locations and environments. Second, we can avoid the risk of overheating, swelling, or exploding of the battery, which can damage the phone or cause safety hazards. Third, we can improve the efficiency and reliability of the phone since it does not suffer from the degradation or self-discharge of the battery.

Using the phone without the battery also has some disadvantages that need to be considered. The phone becomes dependent on an external power source, which may not always be available or reliable. Suppose the power source is disconnected or interrupted. In that case, the phone will shut down immediately, losing unsaved data or ongoing tasks. This can damage the phone's internal components or corrupt the operating system. Furthermore, the phone loses its mobility and portability, as it cannot be moved or carried around without a power cord or adapter. This limits the use cases and scenarios where the phone can be deployed as a cloudlet node, as it requires a nearby power outlet or generator. Moreover, the phone may face compatibility or safety issues with different power sources with different voltage, current, or frequency levels. Therefore causing overheating, short-circuiting, or electrocution of the phone or the user. Thus, a voltage regulator and a surge protector are necessary to prevent such risks.

4.3 RQ3: How feasible is installing Linux-based operating systems on mobile phones?

To answer this research question, we conducted an experiment with a Nexus 5 phone. We tried several solutions to install Linux applications on Android devices, and then we replaced the Android operating system with PostmarketOS. We evaluated the feasibility and performance of each solution and compared them with each other. Here are some of the advantages and disadvantages of each solution:

1. **Linux Deploy**: This app allows us to install a Linux distribution inside a chroot environment on our Android device. We can choose from distributions like Debian, Ubuntu, Fedora, and Arch Linux. We can access the Linux graphical interface through a VNC client. We tried to install it on our Nexus 5, but it did not work. We assumed that we had an old version of Android (4.4.4). One of its advantages is that we can run Linux applications alongside Android applications without modifying the Android system. We can also switch between Linux and Android easily and use the same storage space for both systems. However, we need to root our Android device to use this app, which may void our warranty or cause security issues. We also need to have enough free space on our internal

storage or SD card to install the Linux system. The performance of the Linux system may be slower than a native installation, as it runs on top of the Android kernel and uses emulation.

2. **Lineage OS**: This custom ROM based on Android aims to provide a fast, stable, and secure experience for various devices. It is a fork of the discontinued CyanogenMod project[1], and it supports many features not available in the official Android firmware, such as root access, privacy guard, theme engine, and over-the-air (OTA) updates. Like the Linux Deploy, we tried to install it on our Nexus 5, but it did not work. According to this article[2], the least supported version of Android for Lineage OS is Android 7.1 Nougat, corresponding to Lineage OS 14.1. We later installed it on a Samsung Galaxy A13 with Android version 13 to try it. Using this application, we can enjoy a pure Android experience without any bloatware or unwanted apps. We can customize our device with various settings, themes, and mods. Furthermore, we can benefit from the latest security patches, bug fixes, and new Android versions our device manufacturer does not support. However, we need to unlock our bootloader and flash a custom recovery to install this ROM, which may void our warranty or cause data loss. One of its main drawbacks is that Lineage OS necessitates rooting our device. The process of rooting can be complex, and cancel our warranty. Additionally, some functionalities of our device, such as the camera or fingerprint reader, might need to be supported by Lineage OS. Certain users have also noted Lineage OS stability concerns, which may vary depending on the device.

We could not install several applications such as F-Droid, Termux, and AnLinux because our Nexus 5 was running an old Android version that it had. We installed them on another device, a Samsung Galaxy A13, to test them. We experienced the following benefits and drawbacks of using these applications:

1. **F-Droid**: This app allowed us to access and install free and open-source software for Android without any ads, trackers, or malware. We could also find alternative and niche apps unavailable on the Google Play Store. However, we also missed

[1] https://en.wikipedia.org/wiki/CyanogenMod
[2] https://www.androidauthority.com/lineage-os-devices-908496/

some of the popular and mainstream apps we used, as they were proprietary or depended on Google services. We also had to enable the installation of apps from unknown sources in our Android settings, which could pose some security risks.

2. **Termux**: This app provided a powerful terminal emulator and a Linux environment with various packages, such as bash, coreutils, git, python, and ruby. We could use Linux commands and tools on our Android device without rooting or modifying the system. We could also learn and practice Linux skills and concepts on our device. However, we could only use the Linux command-line interface, not the graphical interface. We also faced some limitations or errors with some commands or tools, as they were not fully compatible or functional on the Android platform.

3. **AnLinux**: This app helped us to install a Linux distribution, such as Ubuntu, Kali, or Fedora, on our device using Termux. We could run a full-fledged Linux system on our mobile device, with more security, stability, and flexibility than Android. We could also run various Linux applications and services compatible with cloud computing. However, we needed enough free space on our internal storage or SD card to install the Linux system. We also needed to use a VNC client to access the Linux graphical interface, which could be slow or laggy depending on the network conditions.

Facing all these challenges and limitations. We decided to replace the Android operating system with PostmarketOS. After installation, the following are the observations we made:

Having mobile phone root access is essential for some users who want to have more control and customization over their devices. By using PostmarketOS, we already have root access by default. Root access allows users to modify system files, install custom ROMs, remove bloatware, tweak performance, and access hidden features. However, rooting also comes with risks and challenges, such as voiding the warranty, losing some functionality, exposing the device to malware, and bricking the device. In our case, after installing PostmarketOS, the battery was partially working; even after several hours of charging, the battery percentage was still showing 46Furthermore, the audio is broken, making calls partially worked. In terms of sensors, only the Accelerometer is partially working. Other than that, we lost the usage of all other sensors.

Running a full-fledged Linux system on our mobile device provides us more security, stability, and flexibility than Android. We can run various Linux applications and services compatible with cloud computing, such as distributed computing, web applications, and web servers. In Addition, we can benefit from the small size and fast performance of Alpine Linux, as well as the long-term support and updates of PostmarketOS. Depending on our preference and device compatibility, we can choose from various user interfaces, such as Plasma Mobile, Phosh, and Sxmo. Furthermore, we can customize our device's settings, such as the kernel, boot loader, and display.

However, postmarketOS is still in development and may not support all the features and functions of our device. We may encounter some bugs, crashes, or compatibility issues. We may also need technical skills and knowledge to install and use postmarketOS on our device. Therefore, postmarketOS is not recommended for casual or inexperienced users who want a stable and reliable OS for their daily use.

4.4 RQ4: Using mobile phones as cloudlet nodes compared to other devices, such as Raspberry Pi or laptops

4.4.1 System setup

Our cloudlet4.2 consists of 4 Linux nodes, one of which acts as a Master and the rest as Slaves. We use a laptop as a controller and a router as a wireless access point to enable SSH communication among the devices. We also use SSH to communicate from the laptop to the devices. For evaluation purposes, we run LINPACK[58], a standard benchmark for HPC systems that typically involve many compute nodes working in parallel. We use an ARM-based LINPACK implementation that relies on an MPI library[59]. MPI is a standard that defines how parallel computations can exchange messages in distributed systems. MPI applications are composed of parallel processes that work on the same problem and communicate via TCP or other protocols.

LINPACK benchmarks are often used to assess a system's floating point performance; even though LINPACK was originally a library for solving linear equations, nowadays, LINPACK is the standard benchmark for the TOP500 list, which ranks the most powerful (known) computer systems in the world.

We use WiFi as the communication link between the nodes, as it is the most feasible option for mobile phones, even though it has higher latency than other links. We

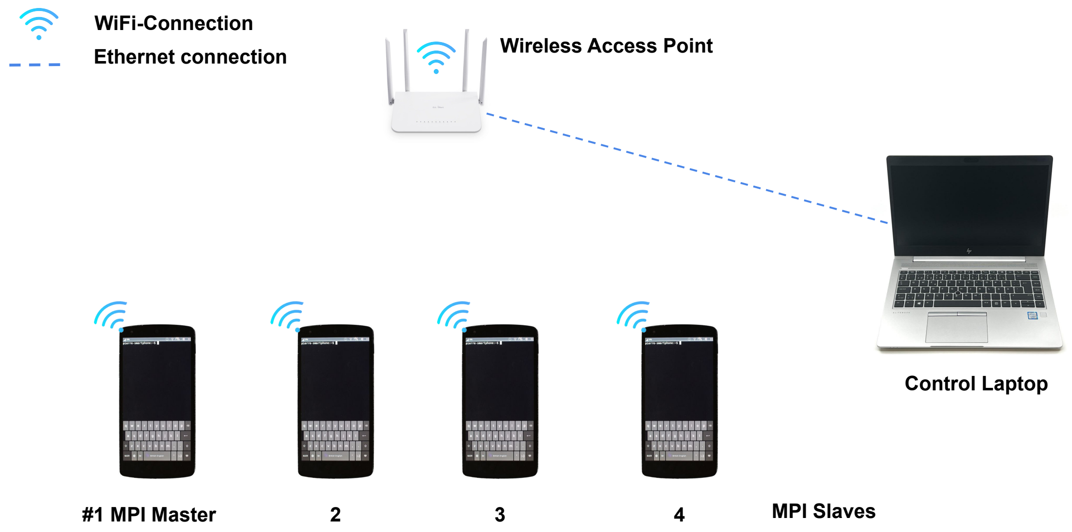

Figure 4.2: Phoenix: Cloudlet System Overview.

perform the LINPACK benchmark with 1 to 4 nodes over the WiFi network. We wrote the HPLinpack configuration files in Python.

4.4.2 Experiment results and discussion

Figure 4.3 shows the averaged results of the LINPACK runs. This experiment shows that with our HPL configuration, a single Nexus 5 reaches 163.34 MFlops on average. This is in line with the performance expected for this architecture when comparing it with the performance reached by the same Android operating system device.

Based on the results in Table 4.1, we can see that mobile phones have the lowest Linpack score among the three devices, even when running a custom operating system (PostmarketOS) that outperforms the default Android OS. Raspberry Pi has about five times higher Linpack score than Nexus 5, and HP EliteBook 840 G5 has about 15 times higher Linpack score than Nexus 5. This shows that the Nexus 5 mobile phones have much lower processing power than Raspberry Pi or laptops, which may limit their ability to handle compute-intensive tasks as cloudlet nodes. Furthermore, after running the FFTE, from the results, we can see that mobile phones have the lowest FFTE score among the three devices, followed by Raspberry Pi and HP EliteBook 840 G5. This

Devices	RAM (GB)	Chipset	OS	MFlops	FFTE (MFlops)	Stream (MB/s)	Random Access (GUPs)
Nexus 5 (1)	2	Qualcomm Snapdragon 800	Linux 3.4.0	162.7	10.6	2.99E-06	0.0070805
Nexus 5 (2)	2	Qualcomm Snapdragon 800	Linux 3.4.0	164.7	10.7	2.95E-06	0.0071716
Nexus 5 (3)	2	Qualcomm Snapdragon 800	Linux 3.4.0	163.6	10.7	2.98E-06	0.0070918
Nexus 5 (4)	2	Qualcomm Snapdragon 800	Linux 3.4.0	164.1	10.8	2.90E-06	0.0007049
Raspberry Pi 4 Model B	4	Cortex-A72 processor	Raspberry Pi OS	837.7	49.7	1.24E-05	0.0432023
HP EliteBook 840 G5	8	Intel Core i5-7200U	Ubuntu 22.04	2468.5	79.7	4.27E-05	0.1033035

Table 4.1: LINPACK performance of different devices systems.

shows that mobile phones have much lower FFT performance than Raspberry Pi or laptops, which may affect their efficiency and accuracy in processing signals and images as cloudlet nodes.

In addition, based on the results, we can see that mobile phones have the lowest Streaming score among the three devices, followed by Raspberry Pi and HP EliteBook 840 G5. This shows that mobile phones have much lower memory bandwidth than Raspberry Pi or laptops, which may cause bottlenecks and delays in data transfer and processing as cloudlet nodes, as shown in Table 4.2. Moreover, the last experiment running the Random Access algorithm shows that the Nexus 5 mobile phones have the lowest Random Access score among the three devices. This shows that mobile phones have much lower random memory access performance than Raspberry Pi, the nearest device we can compare. This may affect their responsiveness and flexibility in handling diverse and complex tasks as cloudlet nodes.

In conclusion, the Nexus 5 has advantages as a cloudlet nodes, such as being widely available, having built-in sensors and cameras, and supporting wireless communication. However, they also have disadvantages, such as limited processing power, memory, and battery life. By comparing mobile phones with other devices, such as Raspberry Pi, we can identify the strengths and weaknesses of each device and choose the best one for our future research work. Based on the results of the four benchmarks, we can see that Raspberry Pi significantly outperforms The Nexus 5 in terms of processing power, FFT performance, memory bandwidth, and random memory access performance. Even when connecting the four devices as a cloudlet, its performance is deficient compared to the Raspberry Pi.

Although the Nexus 5 has lower performance than the Raspberry Pi regarding processing power, it can still be a valuable device for communities with less privilege. The Nexus 5 can be used for educational purposes, such as learning about edge computing, cloudlet nodes, and various benchmarks. The Nexus 5 can also be used for practical

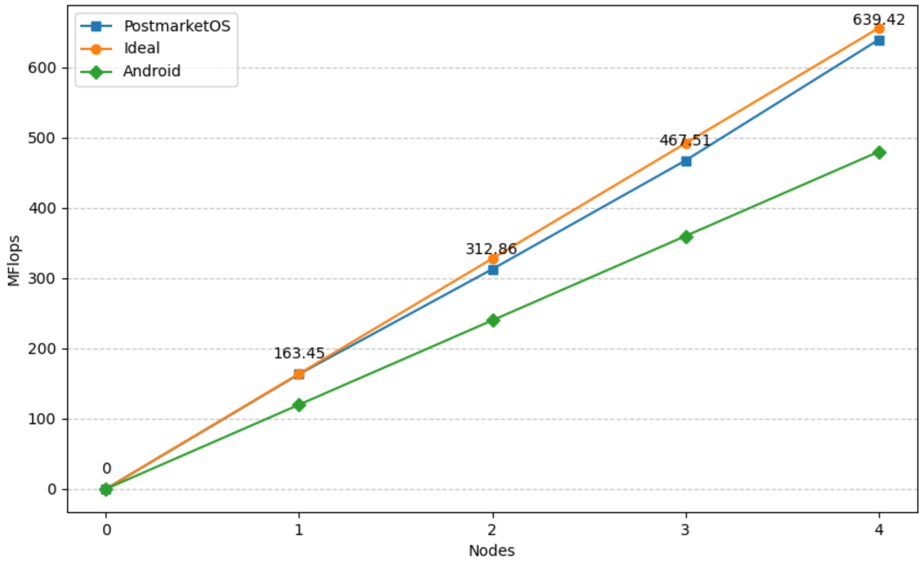

Figure 4.3: Combined computation power (Mega-FLOPS) of clustered smartphones (1 to 4 phones) running Linpack and MPI.

purposes and personal experimentation. Furthermore, the Nexus 5 is more accessible and affordable than the Raspberry Pi, as it is widely available and does not require additional peripherals, such as a monitor, a keyboard, or a mouse. Therefore, it can be a great asset for the communities with less privilege, as it can offer them some opportunities and benefits of using mobile phones as cloudlet nodes.

4.5 RQ5: How can we optimize the performance of the cloudlet?

Our mobile phone-based cloudlets also face some challenges, such as limited performance, reliability, and energy efficiency, which may affect the quality and usability of the Cloudlet. One way to optimize our mobile phone-based cloudlets is to improve their energy efficiency, which can prolong the cloudlet nodes' battery life and reduce the Cloudlet's energy consumption. As we have control of the device root, we disabled the display, leaving the Nexus 5 phones running only the essential services and applications. This can save a lot of energy that would otherwise be wasted on the screen backlight and graphics processing. Another improvement could be adjusting the CPU frequency and voltage of the Nexus 5 phones according to the workload and demand of the Cloudlet, which can balance the trade-off between performance and energy efficiency. Moreover, we can use some energy-saving techniques, such as dynamic voltage and frequency scaling (DVFS) and task migration, to reduce the energy consumption of the Cloudlet further.

Another way to optimize mobile phone-based Cloudlets is to increase their performance, which can enhance the computing power and speed of the Cloudlet. An area of optimization could be increasing the number of nodes (phones), which can increase the Cloudlet's computing power and parallelism. By adding more phones to the Cloudlet, we can distribute the tasks among more nodes and reduce the workload and latency of each node. We can also use some load balancing and scheduling algorithms, such as round-robin, shortest job first, and ant colony optimization, to allocate the tasks to the most suitable nodes and improve the performance of the Cloudlet.

A third way to optimize mobile phone-based cloudlets is to improve their reliability, which can ensure the availability and fault tolerance of the Cloudlet. Reliability is crucial for mobile phone-based cloudlets, as they may encounter some failures and errors due to the mobility and heterogeneity of the nodes. To improve the reliability of the Cloudlet, we can use some replication and backup techniques, such as checkpointing, erasure coding, and redundancy, to store and recover the data and tasks of the Cloudlet in case of node failures. We can also use some fault detection and recovery mechanisms, such as heartbeat, consensus, and voting, to monitor and repair the faulty nodes and maintain the consistency and integrity of the Cloudlet.

A fourth way to optimize mobile phone-based cloudlets is to test their performance in different scenarios and applications, which can help us identify and address the bottlenecks and limitations of Cloudlet. One of the scenarios we tested was the performance of one node serving as a web server. Using the React library, we built and deployed a React application on one of the nodes. The code can be found in our repository in the Appendix. The phone was able to handle the traffic and serve the React application with no noticeable latency. We measured the web server's response time, throughput, and error rate using the Apache JMeter tool. This software can test the performance and functional behavior of various applications. The results showed that the web server

Devices	TCP Window size (KByte)	Interval (sec)	Transfer (MBytes)	Bandwidth (Mbits/sec)
Nexus 5 (1)	16	10.29	69.4	56.6
Nexus 5 (2)	16	10.21	67.9	55.8
Nexus 5 (3)	16	10.45	69.4	55.7
Nexus 5 (4)	16	10.11	53.3	44.2
Raspberry Pi 4	16	10.26	117	95.8

Table 4.2: Network speed test on devices.

Figure 4.4: Repurposed Nexus 5 phones running PostmarketOS.

had an average response time of 0.5 seconds, a throughput of 100 requests per second, and an error rate of 0.01%. These results indicate that the mobile phone-based Cloudlet can provide satisfactory web services for small-scale and low-complexity applications. However, for larger-scale and high-complexity applications, the web server's performance may degrade significantly due to the limited resources and capabilities of the phone. Therefore, we suggest possible solutions to improve the web server performance, such as using a load balancer, a reverse proxy, or a content delivery network (CDN).

4.6 3D printed encasing prototype

The end product of the prototype was the phones, as shown in Figure 4.4. But to make the Cloudlet portable and create a rack for storing the phones, we decided to 3D print a case that will help us achieve several advantages:

1. A 3D printed case can provide physical protection and security for the phones, as

Figure 4.5: Cloudlet 3D rack prototype. (1) the back side, (2) the left side, (3) the front side, and (4) the right side.

they may be exposed to various environmental factors and potential damages.

2. A 3D printed case can improve the thermal management and cooling of the phones, as they may generate a lot of heat when running intensive tasks.

3. A 3D printed case can enable easy installation and removal of the phones, which may need replacing or repairing.

4. A 3D printed case can optimize the space and weight of the Cloudlet, as it can fit the phones in a compact and lightweight design.

5. A 3D printed case can enhance the aesthetic and ergonomic aspects of the Cloudlet, as it can have a customized and user-friendly appearance.

We also made our rack modular to add more components if required, such as more phones, fans, sensors, or LEDs. This gives our cloudlet more flexibility and scalability, as we can customize it according to our needs and preferences. Moreover, our rack has a slide-in solution, meaning each phone is enclosed in a cartridge that allows it to slide into the rack base. Therefore, it makes it easy to insert and remove the phones and connect and disconnect the cables. The slide-in solution also provides a secure and stable position for the phones, preventing them from falling or moving. The combined

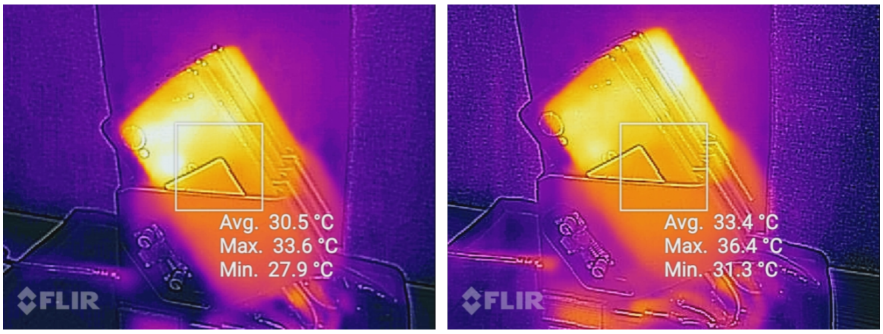

Figure 4.6: This figure shows the heat dissipation of the rack. The left image shows the temperature before the experiment, and the right image shows the temperature after the experiment.

weight of our rack with four phones is 830g, which is relatively light and portable for a cloudlet.

4.6.1 Thermal dissipation

To test the thermal behaviour of the prototype, we run the unixbench benchmark, a suite of tests that measure the performance of a Unix-like system in various aspects, such as CPU, memory, disk, and graphics[1]. We ran the benchmark for 20 minutes on each cloudlet node. We recorded the temperature of the node before and after the benchmark. To measure the temperature, we use the inbuilt FLIR thermal camera from CAT S60 phone, that can capture the heat distribution of the cloudlet nodes. Figure 4.6 shows the result of the experiment, showing the average heat before and after the experiment. We can see that the temperature of the rack increases after running the unixbench benchmark, indicating that the cloudlet nodes generate a lot of heat when performing intensive tasks. We also collected the data containing the change in temperature throughout the experiments, which can be found in the Appendix of this work.

4.7 Summary

In this Section, we present the results of our experiments with the mobile phone-based cloudlet. We repurposed four Nexus 5 phones as cloudlet nodes by installing Postmar-

[1]https://github.com/kdlucas/byte-unixbench

ketOS and running various benchmarks. We compared the performance of the cloudlet nodes with a Raspberry Pi 4 and a laptop using four benchmarks: Linpack, FFTE, Stream, and Random Access. We found that the cloudlet nodes had much lower performance than the other devices in all metrics, even after combining the MFLOPS for the four nodes. This experiment shows that mobile phones have limited processing power, memory, and battery life, which may affect their suitability as cloudlet nodes. We also measured the heat dissipation of the cloudlet nodes stored in a 3D-printed case and found that they generated a lot of heat when performing intensive tasks.

5

Discussions

In this Chapter, we discuss the implications and limitations of our work and what did not work during the experiments. We also provide mobile phone-based cloudlet applications in various domains, such as data self-hosting servers, web hosting, and educational tool support. We hope our work can inspire future research on mobile phone-based cloudlets and contribute to the less privileged communities and the development of edge computing.

5.1 Potential Applications

This Section explores some potential applications of mobile phone-based cloudlets in various domains and scenarios. We discuss how cloudlets can empower communities, provide low-cost microclouds, and support disaster response and recovery. We also highlight some of the challenges and opportunities of using cloudlets in these areas.

5.1.1 Community Empowerment and Resource Optimization

The transformation of e-waste into sustainable mobile phone cloudlets holds the potential to empower communities with limited resources. These cloudlets could serve as localized hubs for running applications, self-deploying to the Internet, and facilitating IoT development. The focus on underwater applications [60][61] further expands the range of potential use cases, catering to environmental monitoring, research, and exploration. For example, mobile phone cloudlets could be used to collect and process data from underwater sensors [62], cameras, and robots. Mobile phone cloudlets could also be used to address some of the challenges and opportunities of underwater environments, such as pollution, climate change, biodiversity, and natural resources. By using mobile phone cloudlets, communities could benefit from the advantages of edge computing, such as low latency, high bandwidth, and privacy preservation, while reducing the environmental impact of e-waste and enhancing the circular economy.

5.1.2 Low-Cost Microclouds for Remote Areas

The deployment of low-cost microclouds in remote areas addresses the challenges of limited infrastructure [63][64]. Repurposed mobile phones could be strategically positioned to create network nodes, enabling data processing and application deployment in regions where traditional cloud infrastructure may be impractical or unavailable.

5.1.3 Disaster Response and Recovery

The ability to power Nexus 5 phones sustainably, even without their batteries, presents a promising solution for disaster-prone areas [65][66]. In scenarios where power supply is scarce or unreliable, these cloudlet nodes could serve as crucial communication and data processing tools for disaster response and recovery efforts [67]. For example, Cloudlet nodes could be used to establish a local network among the disaster survivors, responders, and volunteers and enable voice, text, and video communication. Additionally, they could be used to collect and analyze data from various sources, such as sensors and drones, and provide situational awareness, damage assessment, and resource allocation. Furthermore, Cloudlet nodes could support various applications and services, such as emergency alerts, health care, and education, and improve the well-being and resilience of disaster-affected communities.

5.2 Limitations

In this section, we examine some potential limitations of mobile phone-based cloudlets regarding hardware, software, and energy. We explain how these limitations may affect the quality and usability of the cloudlet.

5.2.1 Hardware Constraints and Processing Power

The limitations associated with repurposed mobile phones include inherent hardware constraints, especially regarding processing power[68]. While mobile phones offer advantages such as built-in sensors and wireless communication[69], they may struggle with demanding computational tasks, impacting the overall performance of the cloudlet. To illustrate this point, we can compare the processing power of mobile phones with other devices, such as Raspberry Pi or laptops, based on benchmark results shown in Table 4.1. These comparisons show that mobile phones have much lower processing power than other devices, such as Raspberry Pi or laptops, which may limit their ability to handle compute-intensive tasks as cloudlet nodes, therefore affecting the quality and usability of the cloudlet, as it may cause delays, errors, or failures in the execution of the tasks. Consequently, optimizing the performance of the mobile phone-based cloudlet by using some techniques, such as load balancing, task scheduling, parallel computing, or cloud offloading, is essential. These techniques can help distribute the workload among the cloudlet nodes, allocate the tasks to the most suitable nodes, exploit the parallelism of the tasks, or offload some tasks to the cloud respectively. By using these techniques, we can improve the performance of the mobile phone-based cloudlet and overcome some of the hardware constraints of mobile phones.

5.2.2 Software Compatibility Challenges

The compatibility of mobile phones with cloudlet services raises challenges related to software[70]. Ensuring that the chosen Linux-based operating system and applications run seamlessly on diverse mobile phone models requires meticulous attention to software compatibility, potentially limiting the range of devices that can be effectively repurposed.

5.2.3 Energy Consumption and Availability

While the proposal suggests sustainable power alternatives, the energy consumption of repurposed mobile phones remains a limitation. Optimizing energy efficiency is crucial for long-term viability, especially in remote areas with limited access to continuous power sources. According to a study done on energy consumption in mobile phones[71], a smartphone consumes just 4 kilowatt-hours of electricity annually, which is relatively low compared to other devices. However, this does not account for the energy consumption of the network infrastructure and the cloud services the smartphone connects to, which can be much higher. Moreover, the energy consumption of the smartphone may vary depending on the usage patterns, the network conditions, and the device's battery health. Therefore, it is essential to measure and manage the energy consumption of the mobile phone-based cloudlet at both the device and system levels.

5.3 Compatibility and Reproducibility for other Devices

One of the main advantages of our project is that it can be applied to a wide range of devices, thanks to the use of PostmarketOS as the operating system. PostmarketOS supports more than 30 fully supported devices and more than 200 devices in the testing phase, including smartphones, tablets, and laptops[1]. This makes installing the operating system on various devices easy, as long as they meet the minimum requirements of having a 3.4 or newer Linux kernel and 512 MB of RAM. PostmarketOS is also in continuous development, and we can expect that more features, such as graphical user interfaces, applications, and drivers, will be added. Therefore, making the operating system more user-friendly and convenient, especially from the installation and update perspectives. Currently, the user needs to go through the same installation process when there is a new update, which can be tedious and time-consuming.

However, there are also some challenges and limitations that we need to consider when applying our project to other devices. One is the compatibility issue with non-booting devices like the Apple iPhone series. Apple products are known for their high security and privacy features, which make them difficult to hack or modify. Apple uses various methods to protect its devices from unauthorized access, such as encryption, authentication, code signing, and hardware security. These methods prevent users from installing a different operating system or modifying the device's firmware unless they have the proper tools and credentials. This issue poses a problem for our project, as we need to bypass the security measures and install PostmarketOS on the device, which may not be possible or legal. Moreover, even if we manage to install PostmarketOS on the device, we may face some functionality issues, such as missing drivers, incompatible hardware, or reduced performance.

[1] https://wiki.postmarketos.org/wiki/Devices

Another challenge we need to address is the reproducibility issue with the battery. As each brand and device model has a different battery configuration and circuit, bypassing the battery will need to be investigated separately for each device. Bypassing the battery means powering the device directly from an external source without using the battery as a buffer or a regulator. This can be tricky, as we need to find the device's correct voltage, current, and polarity and avoid any short circuits or overloads that may damage the device. We also need to find a way to connect the external power source to the device, either by using the existing connectors, such as USB or charging port or by soldering wires to the motherboard or the battery terminals. This may require particular technical skills and tools and may also void the device's warranty.

Finally, we need to consider the hardware issue with the newer phones and how they are built. More recent phones tend to have more complex and compact designs, which make them very difficult to teardown and repair. They often use strong adhesives, proprietary screws, fragile components, and integrated parts, which make it hard to open the device, access the internal parts, or replace the damaged parts. This goes against the trend of self-repairing, which aims to extend the lifespan and functionality of the device by allowing the user to fix it themselves or with the help of a professional. Self-repairing has many benefits, such as saving money, reducing e-waste, and improving security and privacy. However, newer phones make self-repairing almost impossible, forcing users to buy a new device or rely on the manufacturer for repairs. This harms the environment, the economy, and society, creating more waste, costs, and dependency.

5.4 3D Printing Constraints

While 3D printing can accelerate the building of IoT solutions, their deployment in the wild is typically constrained by several contextual and application requirements. Those typically include the environmental conditions (outdoors, indoors), battery autonomy (with solar panels or with direct access to electricity.), and waterproofing (to withstand wind and rain or to be deployed underwater [72]). All these factors need to be considered from the early design stages to provide more robust long-term deployments and guarantee successful operational times. Furthermore, some of the typical 3D printing constraints related to the design are the print size, which depends on the printer and the platform size; the material, which depends on the availability, compatibility, and quality of the printing material; the resolution, which depends on the printing technology, parameters, and post-processing, and the design, which depends on the rules and guidelines for printability and buildability. These constraints affect the printed object's cost, time, functionality, and performance.

5.5 Timeframe for Repurposing the Phones

The timeframe for repurposing the phone depends on various factors, such as the hardware specifications, the software updates, the battery condition, and the user needs. Generally speaking, the older the phone is, the harder it is to repurpose it, as it may have lower performance, less compatibility, and more degradation. According to this study[73], the average lifespan of a smartphone is about 2.5 years, which means that most phones become outdated or broken after that period. However, this does not mean they cannot be repurposed, as some phones may still have functionality and value.

Therefore, the timeframe for repurposing the phone depends on the user's purpose and expectations. Suppose the user wants to use the phone as a cloudlet node, which requires high performance, reliability, and energy efficiency. In that case, the phone may need to be relatively new and well-maintained. On the other hand, if the user wants to use the phone as a backup phone, a webcam, or a baby monitor, which requires less performance but more functionality and compatibility. In that case, the phone may need to be moderately old and functional. A different scenario is if the user wants to use the phone as a dashcam, a DJ device, or a digital photo frame, which requires minimal performance but more creativity and customization, then the phone may be very old and obsolete.

In summary, the timeframe for repurposing the phone is not fixed but relatively flexible and subjective. It depends on the quality and quantity of the phones, the efficiency and effectiveness of the repurposing process, and the user's needs and preferences. Based on our experience, we estimate that the optimal timeframe for repurposing the phone is between one and five years, as it can balance the trade-off between performance and functionality and cost and benefit. However, this is only an approximation, and the actual timeframe may differ depending on various factors. Therefore, we need to plan and manage the timeframe for repurposing the phone carefully, as it may affect the feasibility and scalability of our project.

6

Summary and Conclusion

This thesis explored the feasibility and scalability of using mobile phones as cloudlet nodes for edge computing. We repurposed four Nexus 5 phones by installing PostmarketOS, a Linux-based mobile OS, and running various benchmarks to test their performance. We compared the results with a Raspberry Pi 4 and a laptop and found that the mobile phones had much lower performance in all metrics. We also measured the heat dissipation of the mobile phones and found that they generated a lot of heat when performing intensive tasks.

We discussed some of the advantages and disadvantages of using mobile phones as cloudlet nodes, such as the availability, compatibility, and functionality of the devices and the hardware, software, and energy constraints. We also suggested some techniques to address these challenges and optimize the cloudlet, such as load balancing, task scheduling, parallel computing, cloud offloading, and energy-aware algorithms. We also explored potential applications of the mobile phone-based cloudlet in various domains and scenarios, such as community empowerment, low-cost microclouds, and disaster response and recovery.

We concluded that mobile phone-based cloudlets are a promising approach to providing edge computing services. However, they also face some limitations and challenges that need to be considered and overcome. Moreover, we highlighted the environmental and social benefits of repurposing and recycling mobile phones rather than discarding them as electronic waste. We showed how this cloudlet could help underprivileged students and areas where accessing devices such as Raspberry Pi is very expensive and to be up to date and learn about cutting-edge technologies. We also proposed a way to remove the battery and provide a sustainable power source for the phone to reduce the cloudlet's energy consumption and carbon footprint.

Bibliography

[1] Yu-Gong, X. miao Tian, Y. feng Wu, Zhe-Tan, Lei-Lv, Recent development of recycling lead from scrap crts: A technological review, Waste Management 57 (2016) 176–186, wEEE: Booming for Sustainable Recycling. doi:https://doi.org/10.1016/j.wasman.2015.09.004.
URL https://www.sciencedirect.com/science/article/pii/S0956053X15301136 3

[2] Q. Wu, J. Y. Leung, X. Geng, S. Chen, X. Huang, H. Li, Z. Huang, L. Zhu, J. Chen, Y. Lu, Heavy metal contamination of soil and water in the vicinity of an abandoned e-waste recycling site: Implications for dissemination of heavy metals, Science of The Total Environment 506-507 (2015) 217–225. doi:https://doi.org/10.1016/j.scitotenv.2014.10.121.
URL https://www.sciencedirect.com/science/article/pii/S0048969714015836 4

[3] W. Forum, International e-waste day: Of 16 billion mobile phones possessed worldwide, 5.3 billion will become waste in 2022 (Oct 2022).
URL https://weee-forum.org/ws_news/of-16-billion-mobile-phones-possessed-worldwide-5-3-billion-will-become-waste-in-2022/ 4

[4] M. T. Islam, P. Dias, N. Huda, Waste mobile phones: A survey and analysis of the awareness, consumption and disposal behavior of consumers in australia, Journal of Environmental Management 275 (2020) 111111. doi:https://doi.org/10.1016/j.jenvman.2020.111111.
URL https://www.sciencedirect.com/science/article/pii/S0301479720310380 5

[5] P. K. D. Pramanik, N. Sinhababu, B. Mukherjee, S. Padmanaban, A. Maity, B. K. Upadhyaya, J. B. Holm-Nielsen, P. Choudhury, Power consumption analysis, measurement, management, and issues: A state-of-the-art review of smartphone battery and energy usage, IEEE Access 7 (2019) 182113–182172. doi:10.1109/ACCESS.2019.2958684. 6

[6] A. Fehske, G. Fettweis, J. Malmodin, G. Biczok, The global footprint of mobile communications: The ecological and economic perspective, IEEE Communications Magazine 49 (8) (2011) 55–62. doi:10.1109/MCOM.2011.5978416. 6

[7] A. Danowitz, K. C. McKell, B. Benson, J. Callenes, P. Hummel, R. Randall, Repurposing retired faculty laptops to make engineering more accessible, in: 2019 IEEE Frontiers in Education Conference (FIE), 2019, pp. 1–7. doi:10.1109/FIE43999.2019.9028423. 11

[8] S. Sahni, A. Boustani, T. G. Gutowski, S. C. Graves, Reusing personal computer devices - good or bad for the environment?, in: Proceedings of the 2010 IEEE International Symposium on Sustainable Systems and Technology, 2010, pp. 1–6. doi:10.1109/ISSST.2010.5507714. 12

[9] M. Rahmani, R. Nabizadeh, K. Yaghmaeian, A. H. Mahvi, M. Yunesian, Estimation of waste from computers and mobile phones in iran, Resources, Conservation and Recycling 87 (2014) 21–29. doi:https://doi.org/10.1016/j.resconrec.2014.03.009.
URL https://www.sciencedirect.com/science/article/pii/S0921344914000676 13

[10] H. Riisgaard, M. Mosgaard, K. O. Zacho, Local circles in a circular economy:-the case of smartphone repair in denmark, European Journal of Sustainable Development 5 (1) (2016) 109–124. 13

[11] T. Makov, T. Fishman, M. Chertow, V. Blass, What affects the secondhand value of smartphones: Evidence from ebay: What affects the secondhand value of smartphones, Journal of Industrial Ecology 23. doi:10.1111/jiec.12806. 13

[12] M. Cordella, F. Alfieri, C. Clemm, A. Berwald, Durability of smartphones: A technical analysis of reliability and repairability aspects, Journal of Cleaner Production 286 (2021) 125388. 13

[13] S. Hankammer, R. Jiang, R. Kleer, M. Schymanietz, Are modular and customizable smartphones the future, or doomed to fail? a case study on the introduction of sustainable consumer electronics, CIRP Journal of Manufacturing Science and Technology 23 (2018) 146–155. 13

[14] J. Baxter, I. Gram-Hanssen, Environmental message framing: Enhancing consumer recycling of mobile phones, Resources, Conservation and Recycling 109 (2016) 96–101. doi:https://doi.org/10.1016/j.resconrec.2016.02.012.
URL https://www.sciencedirect.com/science/article/pii/S0921344916300325 13

[15] J. Ylä-Mella, R. L. Keiski, E. Pongrácz, Electronic waste recovery in finland: Consumers' perceptions towards recycling and re-use of mobile phones, Waste Management 45 (2015) 374–384, urban Mining. doi:https://doi.org/10.1016/j.wasman.2015.02.031.
URL https://www.sciencedirect.com/science/article/pii/S0956053X15001348 14

[16] https://independent.academia.edu/SamuelMbuguah, Factors influencing the selection of mobile phone among university students in Kenya — academia.edu, https://www.academia.edu/64605713/Factors_influencing_the_selection_of_mobile_phone_among_university_students_in_Kenya, [Accessed 12-10-2023]. 14

[17] L. Liza, F. Mwaura, The variability in the generation, disposal and recycling of mobile phone e-waste according to social classes in lang°Øata area, nairobi, kenya, Journal of Environment Pollution and Human Health 4 (2) (2016) 42–51.
URL http://pubs.sciepub.com/jephh/4/2/2 14, 15

[18] I. Peñagaricano, Inisghts into mobile phone use, reuse and recycling — mobilemuster.com.au, https://www.mobilemuster.com.au/mobile-phone-use-reuse-and-recycling/, [Accessed 12-10-2023]. 15

[19] S. Janet, Y. Nchawa, "what do i do with my old mobile phones? i just put them in a drawer": Attitudes and perspectives towards the disposal of mobile phones in liverpool, uk, Human Geographies: Journal of Studies and Research in Human Geography 9 (2015) 241–260. doi:10.5719/hgeo.2015.92.8. 15

[20] F. Rasool, You Can Turn Your Old Android Phone Into A Trackpad For Your PC. Here's How - SlashGear — slashgear.com, https://www.slashgear.com/1137188/you-can-turn-your-old-android-phone-into-a-trackpad-for-your-pc-heres-how/, [Accessed 12-10-2023]. 18

[21] J. Fedewa, How to Use Your Android Phone as a Bluetooth Mouse or Keyboard — howtogeek.com, https://www.howtogeek.com/714170/how-to-use-your-android-phone-as-a-bluetooth-mouse-or-keyboard/, [Accessed 12-10-2023]. 19

[22] Yeti - Simplify the control of your smart home — getyeti.webflow.io, https://getyeti.webflow.io/, [Accessed 12-10-2023]. 19

[23] Q. Kong, R. M. Allen, L. Schreier, Y.-W. Kwon, Myshake: A smartphone seismic network for earthquake early warning and beyond, Science Advances 2 (2) (2016) e1501055. arXiv:https://www.science.org/doi/pdf/10.1126/sciadv.1501055, doi:10.1126/sciadv.1501055.
URL https://www.science.org/doi/abs/10.1126/sciadv.1501055 19

[24] J. Raphael, 5 handy Google Fi features you shouldn't forget — computerworld.com, https://www.computerworld.com/article/3325750/google-fi-features.html, [Accessed 12-10-2023]. 19

[25] Root Android: Everything you need to know! — androidauthority.com, https://www.androidauthority.com/root-android-277350/, [Accessed 12-10-2023]. 20

[26] L. LAMPORT, N. LYNCH, Chapter 18 - distributed computing: Models and methods, in: J. VAN LEEUWEN (Ed.), Formal Models and Semantics, Handbook of Theoretical Computer Science, Elsevier, Amsterdam, 1990, pp. 1157–1199. doi:https://doi.org/10.1016/B978-0-444-88074-1.50023-8.
URL https://www.sciencedirect.com/science/article/pii/B9780444880741500238 21, 22

[27] Distributed computing | Parallel Processing, Cloud Computing & Grid Computing — britannica.com, https://www.britannica.com/technology/distributed-computing, [Accessed 13-11-2023]. 22

[28] S. Yi, C. Li, Q. Li, A survey of fog computing: Concepts, applications and issues, in: Proceedings of the 2015 Workshop on Mobile Big Data, Mobidata '15, Association for Computing Machinery, New York, NY, USA, 2015, p. 37–42. doi:10.1145/2757384.2757397.
URL https://doi.org/10.1145/2757384.2757397 23

[29] Y. Liu, J. E. Fieldsend, G. Min, A framework of fog computing: Architecture, challenges, and optimization, IEEE Access 5 (2017) 25445–25454. doi:10.1109/ACCESS.2017.2766923. 23

[30] G. Beniwal, A. Singhrova, "a systematic literature review on iot gateways", Journal of King Saud University - Computer and Information Sciences 34 (10, Part B) (2022) 9541–9563. doi:https://doi.org/10.1016/j.jksuci.2021.11.007.
URL https://www.sciencedirect.com/science/article/pii/S1319157821003219 23

[31] Y. Elkhatib, B. Porter, H. B. Ribeiro, M. F. Zhani, J. Qadir, E. Riviere, On using micro-clouds to deliver the fog (2017). arXiv:1703.00375. 23, 25

[32] E. Sperling, From Cloud To Cloudlets — semiengineering.com, https://semiengineering.com/from-cloud-to-cloudlets/, [Accessed 16-10-2023]. 23

[33] Expanding the Raspberry Pi with PiFace and Pi Rack | Tim Anderson's IT Writing — itwriting.com, https://www.itwriting.com/blog/2362-expanding-the-raspberry-pi-with-piface-and-pi-rack.html, [Accessed 16-10-2023]. 23

[34] Sun Modular Datacenter S20 Product Library Documentation — docs.oracle.com, https://docs.oracle.com/cd/E19115-01/mod.dc.s20/, [Accessed 16-10-2023]. 24

[35] F. Dar, M. Liyanage, M. Radeta, Z. Yin, A. Zuniga, S. Kosta, S. Tarkoma, P. Nurmi, H. Flores, Upscaling fog computing in oceans for underwater pervasive data science using low-cost micro-clouds, ACM Trans. Internet Things 4 (2). doi:10.1145/3575801.
URL https://doi.org/10.1145/3575801 24

[36] K. Saini, P. Raj, Chapter eight - edge platforms, frameworks and applications, in: P. Raj, K. Saini, C. Surianarayanan (Eds.), Edge/Fog Computing Paradigm: The Concept Platforms and Applications, Vol. 127 of Advances in Computers, Elsevier, 2022, pp. 237–258. doi:https://doi.org/10.1016/bs.adcom.2022.02.005.
URL https://www.sciencedirect.com/science/article/pii/S0065245822000341 25

[37] A. Bahtovski, M. Gusev, Cloudlet challenges, Procedia Engineering 69 (2014) 704–711, 24th DAAAM International Symposium on Intelligent Manufacturing and Automation, 2013. doi:https://doi.org/10.1016/j.proeng.2014.03.045.
URL https://www.sciencedirect.com/science/article/pii/S1877705814002914 25

[38] U. Shaukat, E. Ahmed, Z. Anwar, F. Xia, Cloudlet deployment in local wireless networks: Motivation, architectures, applications, and open challenges, Journal of Network and Computer Applications 62 (2016) 18–40. doi:https://doi.org/10.1016/j.jnca.2015.11.009.
URL https://www.sciencedirect.com/science/article/pii/S1084804515002817 25

[39] C. Arun, V. Jaiganesh, Survey on minimizing energy consumption in mobile cloud computing, International Journal of Computer Applications 150 (2016) 5–8. doi:10.5120/ijca2016911471. 25

[40] B. Prasad Rimal, M. Maier, Fog Computing Enhanced Fiber-Wireless Access Networks in the 5G Era, John Wiley & Sons, Ltd, 2019, Ch. 12, pp. 249–271. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781119491590.ch12, doi:https://doi.org/10.1002/9781119491590.ch12.
URL https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119491590.ch12 25

[41] M. Satyanarayanan, A brief history of cloud offload: A personal journey from odyssey through cyber foraging to cloudlets, GetMobile: Mobile Comp. and Comm. 18 (4) (2015) 19–23. doi:10.1145/2721914.2721921.
URL https://doi.org/10.1145/2721914.2721921 26

[42] E. Cuervo, A. Balasubramanian, D.-k. Cho, A. Wolman, S. Saroiu, R. Chandra, P. Bahl, Maui: Making smartphones last longer with code offload, in: Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services, MobiSys '10, Association for Computing Machinery, New York, NY, USA, 2010, p. 49–62. doi:10.1145/1814433.1814441.
URL https://doi.org/10.1145/1814433.1814441 26

[43] M.-R. Ra, A. Sheth, L. Mummert, P. Pillai, D. Wetherall, R. Govindan, Odessa: Enabling interactive perception applications on mobile devices, in: Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services, MobiSys '11, Association for Computing Machinery, New York, NY, USA, 2011, p. 43–56. doi:10.1145/1999995.2000000.
URL https://doi.org/10.1145/1999995.2000000 26

[44] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, A. Patti, Clonecloud: Elastic execution between mobile device and cloud, in: Proceedings of the Sixth Conference on Computer Systems, EuroSys '11, Association for Computing Machinery, New York, NY, USA, 2011, p. 301–314. doi:10.1145/1966445.1966473.
URL https://doi.org/10.1145/1966445.1966473 26

[45] M. Gordon, D. Jamshidi, S. Mahlke, Z. Mao, X. Chen, Comet: code offload by migrating execution transparently, in: Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation, 2012, pp. 93–106. 26

[46] N. Fernando, S. W. Loke, W. Rahayu, Mobile cloud computing: A survey, Future Generation Computer Systems 29 (1) (2013) 84–106, including Special section: AIRCC-NetCoM 2009 and Special section: Clouds and Service-Oriented Architectures. doi:https://doi.org/10.1016/j.future.2012.05.023.
URL https://www.sciencedirect.com/science/article/pii/S0167739X12001318 26

[47] A. Dou, V. Kalogeraki, D. Gunopulos, T. Mielikainen, V. H. Tuulos, Misco: A mapreduce framework for mobile systems, in: Proceedings of the 3rd International Conference on PErvasive Technologies Related to Assistive Environments, PETRA '10, Association for Computing Machinery, New York, NY, USA, 2010, pp. 1–8. doi:10.1145/1839294.1839332.
URL https://doi.org/10.1145/1839294.1839332 27

[48] M. Aazam, E.-N. Huh, Dynamic resource provisioning through fog micro datacenter, in: 2015 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops), 2015, pp. 105–110. doi:10.1109/PERCOMW.2015.7134002. 27

[49] E. Koukoumidis, D. Lymberopoulos, K. Strauss, J. Liu, D. Burger, Pocket cloudlets, SIGPLAN Not. 46 (3) (2011) 171–184. doi:10.1145/1961296.1950387.
URL https://doi.org/10.1145/1961296.1950387 28

[50] K. Habak, M. Ammar, K. A. Harras, E. Zegura, Femto clouds: Leveraging mobile devices to provide cloud service at the edge, in: 2015 IEEE 8th International Conference on Cloud Computing, 2015, pp. 9–16. doi:10.1109/CLOUD.2015.12. 28

[51] H. Li, K. Ota, M. Dong, Learning iot in edge: Deep learning for the internet of things with edge computing, IEEE Network 32 (1) (2018) 96–101. doi:10.1109/MNET.2018.1700202. 28

[52] A. Sathiaseelan, A. Lertsinsrubtavee, A. Jagan, P. Baskaran, J. Crowcroft, Cloudrone: Micro clouds in the sky, in: Proceedings of the 2nd Workshop on Micro Aerial Vehicle Networks, Systems, and Applications for Civilian Use, DroNet '16, Association for Computing Machinery, New York, NY, USA, 2016, p. 41–44. doi:10.1145/2935620.2935625.
URL https://doi.org/10.1145/2935620.2935625 29

[53] F. Dar, M. Liyanage, M. Radeta, Z. Yin, A. Zuniga, S. Kosta, S. Tarkoma, P. Nurmi, H. Flores, Upscaling fog computing in oceans for underwater pervasive data science using low-cost micro-clouds, IEEE Network 4 (2). doi:10.1145/3575801.
URL https://doi.org/10.1145/3575801 29

[54] Y. Lee, L. He, K. G. Shin, Causes and fixes of unexpected phone shutoffs, in: Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services, MobiSys '20, Association for Computing Machinery, New York, NY, USA, 2020, p. 206–219. doi:10.1145/3386901.3389024.
URL https://doi.org/10.1145/3386901.3389024 32

[55] R. Ashima, A. Haleem, S. Bahl, M. Javaid, S. K. Mahla, S. Singh, Automation and manufacturing of smart materials in additive manufacturing technologies using internet of things towards the adoption of industry 4.0, Materials Today: Proceedings 45 (2021) 5081–5088. 45

[56] X. Lopez, K. Afrin, B. Nepal, Examining the design, manufacturing and analytics of smart wearables, Medical Devices & Sensors 3 (3) (2020) e10087. 45

[57] C. Mota, The rise of personal fabrication, in: Proceedings of the 8th ACM conference on Creativity and cognition, 2011, pp. 279–288. 47

[58] The Linpack Benchmark | TOP500 — top500.org, https://www.top500.org/project/linpack/, [Accessed 09-12-2023]. 58

[59] Open MPI: Open Source High Performance Computing — open-mpi.org, https://www.open-mpi.org/, [Accessed 09-12-2023]. 58

[60] H. Flores, N. H. Motlagh, A. Zuniga, M. Liyanage, M. Passananti, S. Tarkoma, M. Youssef, P. Nurmi, Toward large-scale autonomous monitoring and sensing of underwater pollutants (2020). arXiv:2005.09571. 67

[61] H. Flores, N. H. Motlagh, A. Zuniga, M. Liyanage, M. Passananti, S. Tarkoma, M. Youssef, P. Nurmi, Toward large-scale autonomous marine pollution monitoring, IEEE Internet of Things Magazine 4 (1) (2021) 40–45. doi:10.1109/IOTM.0011.2000057. 67

[62] H. F. Rezaei, A. Kruger, C. Just, An energy harvesting scheme for underwater sensor applications, in: 2012 IEEE International Conference on Electro/Information Technology, 2012, pp. 1–4. doi:10.1109/EIT.2012.6220721. 67

[63] J. Panadero, M. Selimi, L. Calvet, J. M. Marquès, F. Freitag, A two-stage multi-criteria optimization method for service placement in decentralized edge micro-clouds, Future Generation Computer Systems 121 (2021) 90–105. doi:https://doi.org/10.1016/j.future.2021.03.013.
URL https://www.sciencedirect.com/science/article/pii/S0167739X21000935 68

[64] M. Selimi, L. Cerdà-Alabern, F. Freitag, L. Veiga, A. Sathiaseelan, J. Crowcroft, A lightweight service placement approach for community network micro-clouds, Journal of Grid Computing 17 (1) (2019) 169–189. doi:10.1007/s10723-018-9437-3.
URL https://doi.org/10.1007/s10723-018-9437-3 68

[65] M. Kurni, K. Madhavi, Enhancing the storage of mobile nodes through ad hoc storage as a service in manets, International Journal of Communication Systems 34 (2020) e4297. doi:10.1002/dac.4297. 68

[66] D. C. Whybark, Co-creation of improved quality in disaster response and recovery, International Journal of Quality Innovation 1 (1) (2015) 3. doi:10.1186/s40887-015-0001-y.
URL https://doi.org/10.1186/s40887-015-0001-y 68

[67] T. Guo, H. Zhang, H. Huang, J. Guo, C. He, Multi-resource fair allocation for composited services in edge micro-clouds, in: 2019 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom), 2019, pp. 405–412. doi:10.1109/ISPA-BDCloud-SustainCom-SocialCom48970.2019.00065. 68

[68] B. Varghese, R. Buyya, Next generation cloud computing: New trends and research directions, Future Generation Computer Systems 79 (2018) 849–861. doi:https://doi.org/10.1016/j.future.2017.09.020.
URL https://www.sciencedirect.com/science/article/pii/S0167739X17302224 69

[69] A. A. Ateya, A. Muthanna, A. Koucheryavy, 5g framework based on multi-level edge computing with d2d enabled communication, in: 2018 20th International Conference on Advanced Communication Technology (ICACT), 2018, pp. 507–512. doi:10.23919/ICACT.2018.8323812. 69

[70] S. Jiang, G. A. Fiore, Y. Yang, J. Ferreira, E. Frazzoli, M. C. González, A review of urban computing for mobile phone traces: Current methods, challenges and opportunities, in: Proceedings of the 2nd ACM SIGKDD International Workshop on Urban Computing, UrbComp '13, Association for Computing Machinery, New York, NY, USA, 2013, pp. 1–9. doi:10.1145/2505821.2505828.
URL https://doi.org/10.1145/2505821.2505828 69

[71] N. Balasubramanian, A. Balasubramanian, A. Venkataramani, Energy consumption in mobile phones: A measurement study and implications for network applications, in: Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement, IMC '09, Association for Computing Machinery, New York, NY, USA, 2009, p. 280–293. doi:10.1145/1644893.1644927.
URL https://doi.org/10.1145/1644893.1644927 69

[72] F. Dar, M. Liyanage, M. Radeta, Z. Yin, A. Zuniga, S. Kosta, S. Tarkoma, P. Nurmi, H. Flores, Upscaling fog computing in oceans for underwater pervasive data science using low-cost micro-clouds, ACM Transactions on Internet of Things 4 (2) (2023) 1–29. 71

[73] Z. Sharifi, S. Shokouhyar, Promoting consumer's attitude toward refurbished mobile phones: A social media analytics approach, Resources, Conservation and Recycling 167 (2021) 105398. doi:https://doi.org/10.1016/j.resconrec.2021.105398.
URL https://www.sciencedirect.com/science/article/pii/S0921344921000057 72

7

Appendix

7.1 What went wrong

In this Section, we describe some of the challenges and difficulties we faced during our project. We also provide some possible explanations and solutions for these problems

1. We installed PostmarketOS with a deprecated version, which caused some features, such as WiFi, not to work. To fix this, we had to edit the kernel, compile the image, and flash it to the phone. This was a time-consuming and error-prone task, as modifying the kernel requires a lot of technical knowledge and careful testing.

2. We installed PostmarketOS on a Samsung Galaxy S3 device without any user interface, just the console version. However, the phone froze, and it kept freezing every time we booted it up. We suspect this was due to hardware or software incompatibility, as the Galaxy S3 is an old and unsupported device. We also found some reports of similar issues with other devices running. A possible solution for this would be to use a newer and more compatible device or install a user interface to help diagnose and troubleshoot the problem.

3. We damaged some of the delicate connectors or wires during disassembly, which caused the phone not to work even after we put back all the components that we removed. We realized we did not follow all the precautions of disassembling the phone. A possible solution for this would be to follow the guidelines and

precautions for tearing down a mobile phone, which can be found in the Appendix of this work.

7.2 Guidelines and Precautions for Disassembling a Mobile Phone

Disassembling a mobile phone is a process that involves opening the device and removing its internal parts, such as the battery, the motherboard, the camera, and the speaker. Disassembling a mobile phone can be useful for various purposes, such as repairing, recycling, or repurposing. However, disassembling a mobile phone also involves risks and challenges, such as damaging the device, losing the parts, or harming oneself. Therefore, it is essential to follow some guidelines and precautions when disassembling a mobile phone, such as:

1. Use the proper tools: To disassemble a mobile phone, you need some tools, such as a screwdriver, a spudger, tweezers, and a suction cup. These tools can help you open the device, remove the screws, pry the parts, and lift the screen. However, you need to use the proper tools that match the size and shape of the device and its parts. For example, you need to use a screwdriver that fits the type and size of the screws, such as Phillips, Torx, or Pentalobe. Using the wrong tools can damage the device or the parts or make them harder to remove.

2. Wear anti-static gloves: When disassembling a mobile phone, you may come into contact with some electrical components, such as the battery, the motherboard, or the connectors. These components can be sensitive to static electricity, which can cause short circuits or damage. Therefore, you need to wear anti-static gloves, which can prevent the buildup of static electricity and protect the components from electrostatic discharge (ESD). You can also use an anti-static mat or wrist strap to ground the static electricity and provide a safe working environment.

3. Keep track of the screws and cables: When disassembling a mobile phone, you may need to remove many screws and cables, which can vary in size, shape, and color. These screws and cables can be easily lost or mixed up, which can make it harder to reassemble the device or cause some functionality issues. Therefore,

you need to keep track of the screws and cables and systematically organize them. You can use a magnetic mat, a plastic tray, or a paper sheet, which can hold the screws and cables in place and label them according to their location and type. You can also take photos or notes, which can help you remember the order and orientation of the screws and cables.

4.

Following these guidelines and precautions, you can disassemble a mobile phone safely and effectively and avoid common problems and mistakes. However, you should also be aware that disassembling a mobile phone may void the device's warranty or violate the manufacturer's terms of service. Therefore, you should only disassemble a mobile phone if you have the permission and the knowledge to do so, and at your own risk and responsibility.

7.3 Source Code

Source code for implementation and all the results of different methods proposed in this thesis are located in the following GitHub repository and links:

- `https://github.com/Blessing92/Phoenix`
- `https://github.com/Blessing92/rate-my-movies`
- `https://drive.google.com/drive/folders/1I1xZinanOFt1-O4DF6JKH59JYkCVXerA?usp=sharing`

7.4 Licence

Non-exclusive license to reproduce thesis and make thesis public

I, Perseverance Munga Ngoy,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,