UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

FIB

A COMPREHENSIVE STUDY ON OPENWISP FOR EVOLVING INFRASTRUCTURE NEEDS

ALICIA PALLAROL ISÁBAL

Thesis supervisor: LEANDRO NAVARRO MOLDES (Department of Computer Architecture)

Degree: Bachelor Degree in Informatics Engineering (Information Technologies)

Bachelor's thesis

Facultat d'Informàtica de Barcelona (FIB)

Universitat Politècnica de Catalunya (UPC) - BarcelonaTech

18/10/2023

Abstract

This thesis explored OpenWISP as a tool for network infrastructures in general and for Hahatay, a northern Senegalese community, in particular. Since OpenWISP, while a powerful platform for network engineers, presents complexities and a non-trivial usage curve, a deep understanding of an organisation's needs was required to assess its value. In response to this gap between the technical application and OpenWISP's capabilities, a guide was developed, offering a walkthrough of this platform and showcasing its functionalities. Furthermore, a Technical Design Document proposing a module marketplace for OpenWISP has also resulted from this work from observed lacks in both centralisation and the community-driven nature of this software.

Resum

Aquesta tesi ha explorat OpenWISP com una eina per a infraestructures de xarxa en general i per a Hahatay, una comunitat del nord del Senegal, en particular. OpenWISP, tot i ser una plataforma potent per enginyers de xarxes, presenta complexitats i una corba d'ús no trivial, per la qual cosa, es requeria una comprensió profunda de les necessitats de les organitzacions per avaluar el seu valor. En resposta a aquesta diferència entre l'aplicació a nivell tècnic i les capacitats d'OpenWISP, s'ha desenvolupat una guia que ofereix un tutorial d'aquesta plataforma tot mostrant les seves funcionalitats. A més, un *Technical Design Document* (Document de Disseny Tècnic) que proposa un mercat de mòduls per a OpenWISP també ha estat el resultat d'aquest treball per la manca observada tant en la centralització com en la naturalesa impulsada per la comunitat d'aquest programari.

Resumen

Esta tesis ha explorado OpenWISP como una herramienta para infraestructuras de red en general y para Hahatay, una comunidad del norte del Senegal, en particular. OpenWISP, a pesar de ser una plataforma potente por ingenieros de redes, presenta complejidades y una curva de uso no trivial, por lo cual, se requería una comprensión profunda de las necesidades de las organizaciones para evaluar su valor. En respuesta a esta diferencia entre la aplicación a nivel técnico y las capacidades de OpenWISP, se ha desarrollado una guía que ofrece un tutorial de esta plataforma mostrando sus funcionalidades. Además, un *Technical Design Document* (Documento de Diseño Técnico) que propone un mercado de módulos para OpenWISP ha sido también resultado de este trabajo por la carencia observada tanto en la centralización como en la naturaleza impulsada por la comunidad de este software.

Acknowledgements

I want to thank my parents, my sister and Pau for supporting me at all times and under all circumstances. They have rowed by my side, and without them, I would not have been able to write this document.

I also want to thank Pilar, Anani and her family. Even with some distance during the elaboration of this document, they always felt close at heart, and with their warmth, they made everything easier.

Moreover, I want to thank Sergio Giménez for his availability, his help, and all the important information and points of view he has given me of Hahatay. And to the network people at KaWo, for their inspiration for the guide.

On a more academical note, I want to thank Leandro Navarro, the thesis director, for his guidance along with the aid provided by Pedro Vílchez and Federico Capoano, the networking, and OpenWISP experts I had the pleasure to meet during this process. They all provided me with fundamental help for carrying out this thesis.

Finally, I want to thank myself, because even at the most intense moments, I did not stop trying.

Index

Abstract **2**
Acknowledgements **4**
1 Introduction **8**
1.1. Background and Context 8
1.2. Problem Statement 9
1.3. Scope and Objectives 10
2 Literature Review and Market Analysis **12**
2.1. Existing Solutions and Technologies 12
2.2. Comparison and Evaluation of Solutions 14
2.3. Identification of Gaps and Opportunities 16
3 Methodology **19**
3.1. Methodological Approach 19
3.1.1. Introduction to the Methodology 19
3.1.2. Research Design 20
3.1.3. Data Collection, Analysis and Validation 20
3.2. Detailed Tasks and Execution 22
3.2.1. Initial tasks 22
3.2.2. Final tasks 26
3.3 Initial budget 30
3.1.1 Personnel costs 30
3.1.2 Generic costs 31
3.1.3 Other costs 33
3.4. Planning and Timeline 35
3.4.1. Initial Schedule and Planning 35
3.4.2. Deviations and Adjustments 36
3.4.3. Impact on Objectives and Costs 37
3.4.4. Final Planning and Timeline 39
4 Sustainability report **41**
4.1. The Project Put into Production (PPP) 41
4.2. The Lifespan 42
4.3. Risks 43
4.4. Conclusions 44
5 OpenWISP and Organisations: Customisation and Network Optimisation **45**
5.1. Specific needs of Hahatay 46
5.2. Customisation: User Roles and Network Policies 46
5.2.1. User roles 46
5.2.2. Authentication and Access 48
5.2.3. Monitoring and Data Collection: 49
5.2.4. Network Policies 50
5.3. Optimisation: Channel Selection, Traffic Management, and VLAN Implementation 51
5.3.1. Channel Selection 51

5.3.2. Traffic Management 52
5.3.3. VLAN Implementation 53
5.4. Best Practices and Challenges 55
6 Ubiquitous Access and User-Centred Design 57
6.1. Ensuring Comprehensive Network Coverage 58
6.2. Design of the User Guide 61
6.2.1. Outline of the guide 62
6.2.2. Testing environment to familiarise with OpenWISP: 63
6.2.3. Other considerations on the guide 64
7 Evaluations, Iterative Improvements and Proposal 65
7.1. Preliminary Evaluations and Lessons 65
7.1.1. Objectives of Simulations 65
7.1.2. Overview of Results 65
7.1.3. Lessons Learned 67
7.2. Cost and Benefit Analysis 68
7.3. Adapting and Refining the Network Over Time 71
7.3.1. Key Performance Indicators (KPIs): 72
7.3.2. Evaluations and feedback 73
7.3.3 Other considerations 73
7.4. OpenWISP Enhancement Proposal (OEP) 74
7.4.1. Detailed Enhancement Proposal 74
7.4.2. Potential Impact and Importance 75
8 Conclusion and Future Directions 77
8.1. Key Findings 77
8.2. Recommendations for Further Work 78
8.3. Closing statements and personal conclusion 79
Bibliography 80
Appendix A 86
Appendix B 127

List of figures:

1. Gantt diagram with the initial planning 35

2. Gantt diagram with the final planning 39

3. Graph representing Hahatay's areas 68

List of tables:

1. Comparison of existing solutions 14

2. Annual and hourly salary for the different roles in this project 30

3. Hour distribution for the different task groups and the responsible roles for each one 30

4. Summary of amortisation of hardware components required in this thesis 31

5. Personnel and generic costs 32

6. Incidental budget calculation 33

7. Summarization of contingencies and incidents 34

8. Budget summary 34

9. Hour distribution for the different task groups and the responsible roles for each one according to the final planning 37

10. Summary of amortisation of hardware components required in this thesis according to the final planning 38

11. Graph representing Hahatay's areas according to the final planning 38

1 Introduction

1.1. Background and Context

This document is a Bachelor's Thesis for a Computer Engineering Degree, specialisation in Information Technologies.

This thesis is done in the Facultat d'Informàtica de Barcelona of the Universitat Politècnica de Catalunya (UPC), and Leandro Navarro Moldes is the supervisor.

The primary motivation behind this thesis is that there are many inequalities in this globalised world. One of them is the existence of a digital divide. The digital divide is defined by Tien and Fu (2008) as a "technology capacity gap that exists between those who are 'information-rich' and those who are 'information-poor'".

The divide manifests in two primary challenges: the infrastructure and the literacy in different economic communities (Chetty et al., 2018). For this reason, the different non-governmental organisations (NGOs) fighting to end these inequalities follow different strategies.

Some organisations, such as Close the Gap, focus on the infrastructure issue as they work on bridging the digital divide (Close the Gap, n.d.). They offer donated second-hand technological devices to different projects ranging from educational to medical, and social projects.

There are other organisations with a different strategy for fighting the divide, more centred on the second problem above mentioned such as the NGO Team4Tech. Their projects help

teachers and staff gain digital literacy so that they can ultimately transfer the knowledge to students (Team4Tech, n.d.).

At UPC's Center for Development and Cooperation, several associations and initiatives are working on the earlier described matters. One of these associations later to be discussed is AUCOOP (*Associació D'Universitaris per la Cooperació* in Catalan, which translates as University Students for Cooperation Association).

1.2. Problem Statement

Both this thesis' author and director are members of AUCOOP. This association works with different entities and NGOs that either propose different issues that the association translates into projects or directly propose a project in which the association could take part.

The projects are in developing areas both local and international and can be related to schools, hospitals or even bigger communities like whole towns. This association works on both the main digital divide issues: the infrastructure, as they bring hardware components of all kinds, upgrade and set up of networking systems, etc.; and the literacy, as they provide training for teachers and staff involved in the projects.

One recurrent setback to these projects' development is related to the arduity and sometimes complexity behind the Wi-Fi coverage and connectivity management, set up and monitoring. For example, there is an important area to cover or there are many users whose credentials need administration.

In the case of Hahatay, a northern Senegalese community where AUCOOP has a project, their infrastructure would benefit from having more network monitoring tools. Moreover, they would benefit of a user management systems in order to avoid usage abuses.

In addressing these challenges, open-source software emerges as a key solution. Why? Open-source solutions offer several advantages including cost-effectiveness, flexibility and adaptability, community support, and an inherent transparency that fosters trust. Such attributes are especially vital when resources are limited and adaptability is essential.

And when discussing open source solutions, OpenWISP emerges as a promising tool. Originating in 2008 from the Italian project ProvinciaWi-Fi (Barcelo et al., 2012), this open-source network management system offers capabilities that could simplify the tasks associated with Wi-Fi management. The goal of this project is to deeply explore the potential and intricacies of OpenWISP in cooperation contexts specifically, emphasising its potential benefits and applicability.

1.3. Scope and Objectives

The endgame of this project, making a comprehensive Study on OpenWISP for Evolving Infrastructure Needs is a rather general aim. For this motive, it is crucial to define a scope with smaller objectives and sub-objectives, specific requirements and potential risks so that there is a better-centred point. The first scope we will be fixing is that the central organisation looked up is the above-mentioned Hahatay community.

The theoretical objectives of this thesis are:

- Understanding the needs of an organisation setting up Wi-Fi with any network infrastructure. Thus, the sub-objectives are:

    - Gain perspective on different Wi-Fi infrastructure contexts within organisations and cooperation projects. These often include challenges such as coverage issues in physically challenging terrains, network congestion with high user loads, security concerns in environments with limited IT expertise, hardware limitations due to budget constraints, and the complexities of maintenance and troubleshooting in remote locations.
    - Learn about common hardware used in cooperation projects or at the Hahatay community with a particular emphasis on OpenWISP.
    - Analyse the information to extract specific needs and issues.
- Finding meaningful solutions to the previously mentioned needs:
    - Explore different open-source software related to bridging the digital divide other than OpenWISP.
    - Determine the different strategies to solve most of the previously identified issues by using OpenWISP, if possible.

The practical objective of this thesis is, for each solution found in the theoretical part, to make sure that it is a feasible option for a cooperation project; thus the sub-objectives are:

- OpenWISP architecture analysis: Familiarise and document the core components and structures of OpenWISP to form a foundation for subsequent tasks.
- Configuration and troubleshooting: Attempt to enforce configurations on devices, detailing the process and challenges faced.
- Design principles for network setup: Using the foundational knowledge of OpenWISP, propose potential setups and configurations that might optimise network performance and ensure wide access.
- Recommendations and future framework goal: Offer insights derived from the experience for future endeavours, alongside a suggested framework for technical evaluations and simulations.

Based on the insights and solutions derived from these objectives, this thesis will provide a guide to optimise Wi-Fi infrastructure using open-source tools tailored for cooperation projects.

2 Literature Review and Market Analysis

2.1. Existing Solutions and Technologies

There are many initiatives focused on the digital divide related to networking, specifically Wi-Fi connectivity, but also many initiatives attending different aspects of the Wi-Fi infrastructure trying to facilitate different processes.

The Bottom-up Broadband for Europe project (Barcelo, 2014), for example, serves as a foundational initiative in the journey to combat the digital divide. While not a direct technical solution, it aimed to champion the need for open/free networks, laying the groundwork and emphasizing the importance of sophisticated tools and systems to truly eradicate connectivity disparities. Its ethos is reflected in subsequent developments and solutions like Guifi Net.

It is worth mentioning Guifi·Net (Fundació Guifi·net, n.d.) since it is a local initiative, however, while it serves as a prominent example of community-driven networks, the focus of this thesis narrows down to software platforms that offer comprehensive network management capabilities.

The first technical solution under consideration is LibreMesh: “a modular framework for creating OpenWrt-based firmware for wireless mesh nodes” (LibreMesh, n.d.). This is a

project originated by a group of free network activists back in 2013; the intention was to create a common solution for the deployment of free mesh networks. Such networks are particularly valuable in areas where traditional infrastructure might be lacking or too expensive to deploy.

Another solution is the already mentioned OpenWISP, the cornerstone software of this thesis. This software platform automates and facilitates network management (OpenWISP, n.d. -d). It provides some interesting tools as it uses open standards, software tools, and low-cost hardware and was intended to provide users with Internet access.

A more well-known technology is Eduroam, a well-known service among research and education communities around the world (NSRC, 2017) as a roaming access provider. It makes credential management seamless from both the users' and the institutions' points of view.

Yet another example is MikroTik RouterOS, a Linux-derived operating system tailored for router-based computers, emphasising user-friendliness and adaptability in network administration. Originating in 1995 for wireless ISPs, it has grown in global prominence. It is a versatile tool that offers many features with stable and quality-controlled routing capabilities. However, while it offers a myriad of capabilities, it operates on a licensable model, segmented into different "Levels", requiring payment for full access (Sumarno et al., 2019).

The presented solutions differ significantly in their scope and application. While each offers unique advantages, it is essential to understand their inherent features, usability, cost or

pricing, scalability, flexibility and the level of community support they receive. These aspects will be crucial when considering their suitability for cooperation projects.

2.2. Comparison and Evaluation of Solutions

As previously discussed, it is very important to be aware of the differences among the existing solutions. To illustrate and explain these differences, we find the following table, Table 1, where we will deep dive into the strengths, weaknesses, and unique selling points of each software tool. Guifi Net and Bottom-up Broadband for Europe are more of an infrastructure-based tool and a foundational initiative, for this reason they are excluded from this comparison:

	LibreMesh	OpenWISP	Eduroam	MikroTik RouterOS
Main features	- Network segmentation - Layer 2 roaming in set areas - Smart gateway selection with redundancy - Compatibility for various scenarios - Single firmware for all network types	- Network management: new nodes, different network configurations, mesh networks and wireless access points and firmware upgrades - Connectivity and Security: creation of VPN tunnels, RADIUS authentication, captive page - Monitoring and Notifications: low maintenance, web and email notifications, network topology visualization	- Federated authentication - Secured network - Single SSID across participating institutions - Wide geographic coverage	- Routing firewall - Wireless access point - Backhaul links - Hotspot systems - VPN servers - Bandwidth management - Application filtering

Usability	- Open source - Modularity - Feature complexity - Need for deep technical knowledge	- Open source - User-friendly interface - Feature complexity - Need for deep technical knowledge	- Easy to connect - User-friendly interface - Reliable performance - Initial set-up complexity - Inconsistent availability in different institutions	- Ease of use - Versatile and multi-functional - Stability and quality control - Licensing model
Cost	Free	Free	Free	6 levels with different functionalities available each. Ranging from free at level 0 and 1 to $250 at the top level
Scalability	- Can grow with added nodes. - Suitable for decentralised networks.	- Enables the managing of multiple access points and routers. - Designed for varying locations and scales.	- Global roaming service scalable across institutions worldwide. - High scalability with multiple institutions connected	- Supports small to large networks. - Ability to handle thousands of users.
Community support	- Active community of enthusiasts and developers. - Lacks commercial support structure.	- Supported by an active community. - Offers commercial support options.	- Supported by academic institutions globally. - Centralised and regional operational support centres.	- Active user forum and Wiki. - Official training and consultancy services available.

Table 1[1]: Comparison of existing solutions.

Let's delve into a deeper comparison to evaluate each solution:

- Eduroam's strength lies in its federated authentication system, making it seamless for academic users to access Wi-Fi in participating institutions globally. This creates

[1] Source: own elaboration, content created according to LibreMesh (n.d.-a and n.d.-b), OpenWISP (n.d.-a, n.d.-b and n.d.-c), Incommon (n.d.), NSRC (2017), Sumarno et al. (2019) and MikroTik (n.d.).

a robust, collaborative global Wi-Fi network. However, its specific focus on academic institutions may limit its application in diverse contexts, especially in areas without solid academic infrastructures such as the one at Hahatay.

- LibreMesh and OpenWISP serve different primary purposes, even if there is some overlap. LibreMesh focuses on creating a mesh network with seamless roaming, smart gateway selections, and more. Its strengths are in deploying mesh networks in challenging environments, like rural areas or places with sparse infrastructure. OpenWISP, on the other hand, is more about network management. It is about making it easy to manage, configure, and monitor networks.
- It is necessary to acknowledge the robustness of MikroTik RouterOS's comprehensive features that place it on par with solutions like OpenWISP. The fact that it is not open source can be considered both an advantage and a disadvantage. On the one hand, commercial solutions often come with more structured support and possibly more polished user experiences. On the other hand, the essence of this thesis leans towards open source due to its adaptability, customisability, and the philosophy of community-driven development.

While these existing solutions offer an array of features and cater to varied needs, it is evident that each solution has its strengths, niches, and limitations. Identifying gaps within these technologies can pave the way for enhanced or hybrid solutions that address specific challenges in the context of bridging the digital divide.

2.3. Identification of Gaps and Opportunities

Among the various solutions explored, OpenWISP stands out for its open-source nature, comprehensive network management capabilities, and adaptability to diverse scenarios. The choice of focusing on OpenWISP is deliberate. In the panorama of available solutions, OpenWISP's unique blend of features, open-source nature and adaptability positions it as a

promising tool to address connectivity challenges. While other solutions present their own merits, OpenWISP's features and philosophy are particularly aligned with the goals of bridging the digital divide in a community-driven manner.

While each solution possesses its unique strengths and limitations, OpenWISP, being central to this thesis, showcases some evident gaps that merit attention:

1. Technical expertise requirement: While OpenWISP is powerful, it might demand a certain degree of technical proficiency for installation, configuration, and ongoing management.
2. Integration and compatibility: OpenWISP might not seamlessly integrate with all hardware types or existing networking solutions.
3. Scalability challenges: While OpenWISP can handle multiple access points and routers, it might face challenges as the network scales or when integrating with other network architectures.
4. Community and support: As with many open-source solutions, real-time support might be lacking, relying heavily on community forums or available documentation.
5. Customisation and features: There might be features that specific communities or setups need that OpenWISP currently does not offer.
6. Knowledge transfer and continuity: This main point relates to the first one, because of the technical knowledge required, organisations or communities might face challenges if their primary OpenWISP operator leaves or is unavailable.

As with any technology, each identified gap presents a canvas for innovation. For OpenWISP, these gaps translate to opportunities that can redefine its applicability and efficiency:

1. Developing comprehensive training programs or user-friendly documentation tailored for individuals without a technical background could help to bridge the knowledge divide.

2. Creating plugins or extensions that make OpenWISP compatible with a wider range of devices. A collaboration with hardware manufacturers or the community to ensure smoother integration could solve this.
3. Enhancing the scalability features of OpenWISP. Research a hybrid model where OpenWISP can work in tandem with other solutions when scaling could help with this issue.
4. Establishing a dedicated support team where entities can offer professional support for OpenWISP deployments.
5. Creating a platform within OpenWISP where developers can offer custom modules with their own features, enhancing the system's overall capability. A marketplace or a feature-request forum could be beneficial.
6. Developing a mentorship or training program to ensure that knowledge about OpenWISP operation and management is passed on and shared among multiple individuals in a community or organisation. This ensures continuity and reduces dependency on a single individual.

Throughout this chapter, we have explored a myriad of software tools and solutions that aim to address the digital divide, emphasizing their unique features and potential shortcomings. OpenWISP, with its open-source nature and comprehensive network management capabilities, emerges as a promising contender in this space. Given the scope and objectives of this research, the following chapters delve deeper into the possibilities presented by the identified gaps and opportunities associated with OpenWISP, aiming to further its potential as a tool for bridging digital disparities.

Specifically, the degree of technical proficiency required for installation, configuration, and management is tackled with a guide. And to elaborate it in the most relevant way for any organisation, the different needs OpenWISP serves are discussed.

3 Methodology

3.1. Methodological Approach

3.1.1. Introduction to the Methodology

Kanban was the chosen methodology for this project as it is a visual aid and the objectives set can be achieved with some parallelising. Moreover, as this is a research project where there can be multiple paths this methodology offered enough flexibility.

This methodology (David James Anderson, 2010, pp. 13–16, 113–120, 139–144) is based on cards, those cards represent tasks or, in general, pieces of work. The cards are placed in different columns according to their state. For this project, the defined states are:

- To Do: The list of tasks pending. There are no limits on this column.
- Development: Ongoing tasks. This column has a limit of 6 cards.
- Quality assurance: Finished tasks that need a review to be declared as done. This column has a limit of 3 cards.
- Done: Tasks that are finished and up to the standards of the project so that the work related to them can be included in the documentation. This column has no limits.

The limits on columns are key to implementing this method as restrictions on cards' states before being done help control multitasking in the possible parallelisation mentioned above.

To apply this methodology, we used the app Trello with its board for the project with the columns mentioned earlier as lists in Trello jargon. This application allows adding tasks and subtasks inside them, so it was easy to monitor the performance and attainment of the goals.

3.1.2. Research Design

The nature of this thesis, which focuses on OpenWISP in organisations, needed an in-depth understanding of the underlying needs, challenges, and contexts that numbers alone could not provide. As such, a qualitative research design was chosen for the following reasons:

1. Depth and Detail: Qualitative research allows for a deeper dive into the complexities of setting up and customising a network system, understanding not just the "how" but also the "why" behind each decision and challenge.
2. Contextual Understanding: Given the unique context of Hahatay, it is crucial to capture the specific needs, preferences, and challenges that the stakeholders (students, staff, administrators and other community members) face. Qualitative methods, like interviews or observations, can provide such context-rich insights.
3. Iterative Nature: The process of implementing a network system is not linear. Feedback loops, testing, and revisions are integral. Qualitative research, being flexible, allows for such iterative processes and helps in capturing the evolution of the research and the reasons for any changes or adjustments.
4. Capturing Subjectivity: Understanding user experiences, satisfaction levels, and preferences is inherently subjective. Qualitative research is adept at capturing such subjectivities, making it an ideal choice for a user-centric project like this.

3.1.3. Data Collection, Analysis and Validation

As a research project, it is worth mentioning the general data gathering and processing procedures.

As per the sources of data:

- OpenWISP Documentation: Comprehensive review and extraction of relevant information from OpenWISP's official documentation, focusing on customisation options, modules, user role definitions, and network optimisation strategies.

- Academic Journals and Papers: To support the study, peer-reviewed journals and papers were consulted. These provided insights on network optimisation strategies, ubiquitous access, user-centred design principles, and more.
- Involved parties: I had direct information of the current infrastructure and its needs directly from people directly involved in Hahatay by videoconferences.

As per the data analysis:

- Literature Review: Systematic review of academic journals, papers, and other credible sources to gain insights into existing strategies, best practices, challenges, and trends in the realm of network systems and optimisation.
- OpenWISP Documentation Study: Detailed study of OpenWISP's documentation to comprehend its functionalities, customisation options, and best practices. Special emphasis was placed on areas directly relevant to the needs of communities, such as user role definitions and network usage policies.

To mention specific techniques for this data analysis:

- Thematic Analysis: Data, especially feedback and observations, was analysed to identify recurring themes or patterns. This helped in understanding common challenges, user preferences, and areas of improvement.
- Document Review: All documented data, be it code, reports, or visual documentation, was periodically reviewed to obtain insights, learn from past experiences, and inform future actions.

As per the validation, a GitHub repository was maintained for all developed code. However, all the writing and reporting was stored on Google Drive, including theoretical frameworks, design plans, user feedback, and other relevant data. For the writing process, OpenAI's ChatGPT was consulted to ensure polished content and the best written expression possible.

Additionally, weekly meetings were scheduled with the project director to review the progress and decide the approach for the goals and tasks.

3.2. Detailed Tasks and Execution

In this section, there is a description of all tasks that carried out throughout the project. However, since the project changed from the initial planning, to aid coherence, we discuss this in two sections, one for the initial tasks and the other one for the actual tasks carried out and the necessary adjustments.

Tasks are grouped and have subtasks so it is easier to temporally plan. Additionally, if in the description is not specified the dependencies of precedence it is implied that there are not any. Finally, some tasks or groups of tasks are not changed, those are only detailed in the initial tasks section.

3.2.1. Initial tasks

As a thesis, since it is not a small project, the first tasks are related to **project planning** (PP). This first set of tasks require a computer with an Internet connection, a Google account to document it and meetings with the thesis director.

- **PP1 -** Contextualisation and project scope: the definition of the project and its scope with its proper contextualisation will take 30 hours.
    - Gaining familiarity with the terms and specific vocabulary of the field.
    - Reaching an agreement on the focus of the project: schedule a meeting to discuss the scope of the project with the thesis director.
    - Studying the current state-of-the-art: investigating different projects similar to this very one.

- **PP2 -** Temporal planning: the definition of tasks and situating them over a Gantt diagram will take 25 hours. Aside from the previously mentioned resources, it will also need the software Asana and Instagantt for the Gantt diagram.
- **PP3 -** Budget and sustainability: budgeting and analysing the sustainability of this project will take 30 more hours.
- **PP4 -** Integration: it is needed to merge the previously described tasks coherently. As we want to be thorough, this will take 20 hours and it will require that the 3 previous tasks are already completed.
- **PP5 -** Meetings: a weekly meeting is scheduled with the thesis director. Since there are 18 weeks, this task will take 20 hours taking into account that in some moments the meetings can be more frequent.

This project needs a research base to find meaningful solutions, for this reason, there is an essential set of tasks focused on the **groundwork** (GW). All groundwork tasks share the need for a computer and Internet access.

- **GW1 -** Understanding the needs of AUCOOP in general and in Senegal specifically with the Wi-Fi set-up. This task with its subtasks will take 30h.
    - GW1.1 - Interviewing at least two members of an organisation such as the AUCOOP association. With participation in the Hahatay project if possible. This needs an arrangement with the people at the association.
    - GW1.2 - Investigating the hardware and software used in past projects.
- **GW2 -** Assessing the strengths and weaknesses of the software and hardware used by the association in the past. Analysing the information to extract the potential of the current infrastructure or the need for improvement of the current state. This task will take 25h. To start this task it is required that GW1 is initiated, not necessarily finished.

- **GW3** - Research solutions to the previously mentioned needs. This task and its subtasks will take 40 hours since this task requires a deep investigation. This task needs GW1 and GW2 to be finished since we need the conclusions of that part to put the focus properly on this task.
    - GW3.1 - Study different open-source software to assist with those needs.
    - GW3.2 - Determine three possible solutions that can be implemented.

Once we have the solutions defined, we will start with the hands-on part. This practical part will involve repeating the same task iterations for each of the 3 different solutions. This set of tasks related to **experimentation** (EX) with their subtasks will also require a computer and also an access point, and some other hardware as well as software, to be determined, depending on the solutions found:

- **EX1 -** Designing and preparing a demonstration simulating a real scenario where the solution defined in GW3 can be used, for this reason, the first iteration of this task can start once a solution is determined but not necessarily all of them, except for the last iteration that needs the last solution of GW3 and therefore needing the task GW3 to be finished. Since solutions may differ and are not yet defined, preparations can vary, including hardware acquisition or requesting access to specific devices for specific software. Each iteration is estimated to require 30 hours.
- **EX2 -** Carrying out the demonstration of the solutions identified in EX1. Due to the potential variability of solutions, specific subtasks and resources cannot be specified, but an access point is expected to be necessary since this is a Wi-Fi project. The dependencies in this task come from the different iterations. Each iteration will be identified as *task tag_iteration number*, so the first iteration of EX1 would be EX1_1, for EX2 the dependencies are at the iteration level so for the second solution EX2_2 needs that EX1_2 is finished. For each iteration, this task is expected to need 30h.

- **EX3 -** Evaluating if it is a viable solution taking into account the metrics of simplicity in deployment in comparison with the previous and cost/need of maintenance. Evaluate the quality and impact of the solution. The different solutions might need different resources for the measurements of this task. The estimation is that for each iteration this task will take 20h. The dependencies follow the model of EX2, so EX3_1 needs that EX2_1 is finished to start.

Since there are 3 iterations, we have that EX1 will take 90 hours with EX1_1, EX1_2 and EX1_3 lasting 30 hours each. For EX2 the total is 105 hours and for EX3 60 hours.

Another important task is the **conclusion** drawing (C):

- **C -** Conclusion: analysing the results from the experimentation to summarise the study. It will require a computer, with Internet access and it is dependent on the definitive ending of all experimental iterations so we have the broader picture. This task will take 15 hours.

There is one task that can be done during the whole project as it has no dependencies and it is key for the success of this project, the **documentation** (D) task:

- **D -** Documenting all processes and events and integrating all collected information. It will take 65 hours and the resources required are a computer and Internet access to write the document in Google Drive.

Finally, a task that will not be seen on paper is the **oral defence** of the thesis, which will take place in June. This task will take the remaining hours we have left to dedicate, and the resources needed are information resources such as who is to be the Tribunal, to prepare for questions. We will exclude this part from the upcoming tables and Gantt diagram since, as mentioned, it will not be documented.

3.2.2. Final tasks

From the initial tasks we maintain the whole set of the **project planning** (PP). But we will extend it with one additional task:

- **PP6: -** Stakeholder Engagement and Feedback Collection: Schedule and conduct more meetings with external stakeholders related to Hahatay to monitor the progress and check new findings. Estimate: 10 hours.

As for the **Groundwork** (GW), it is crucial to understand the context of the community at Hahatay, for this reason, we are maintaining **GW1** and **GW2**, however, **GW3** tasks changed and **GW4** has appeared:

- **GW3 -** Familiarise with OpenWISP's architecture and find links to the most important weaknesses at Hahatay's Wi-Fi infrastructure. This task and its subtasks are planned to take 20 hours.
    - GW3.1: Checking the demo offered at OpenWISP's web page and familiarising with the modules and artefacts that compose the platform.
    - GW3.2: Studying its functionalities and matching them with the needs at Hahatay.
- **GW4 -** Immersion in an OpenWISP's Practical Deployment. Learn in a practical environment about OpenWISP. Participate in a project where OpenWISP is being used. This can be done with the celebration of the 2023 Battle Mesh at Calafaou (Barcelona), which is an annual convention on free networking (Wireless Battle Mesh, n.d.). This can provide an insight on the capabilities of OpenWISP. The task and its subtasks are estimated to take 30 hours.
    - GW4.1 - Active Participation: engaging hands-on in the OpenWISP setup process at some preparatory activities of the convention. This provided a real-world understanding of its deployment, challenges, and nuances. This task requires prior basic knowledge of OpenWISP but does not necessarily

need in-depth experience. To participate it is possible that a router and an Ethernet cable are required.

    - GW4.2 - Networking and Interactions: making connections with professionals and enthusiasts in the free network realm, specifically related to OpenWISP. This can provide first-hand insights from those experienced with free networks and OpenWISP.

Moreover, we should take into account that this work although focused on Hahatay should be available to other institutions or communities. For this reason, we also have within Groundwork:

- **GW5 -** Understanding Local User Profiles: Gather insights into the typical user profiles in Hahatay and also in a general setting. Their digital behaviour, the devices they use so that the needs can be tailored to the users. This task is planned to take approximately 10 hours.
- **GW6 -** Research on network policies and link them with the previous user profiles. Figure out how they can accommodate the needs of Hahatay. This task is planned to take take approximately 15 hours.

The hands-on section is the most different one. Experimentation (EX) has changed to **OpenWISP Functionality Exploration** (OE) since a deep dive into OpenWISP's capabilities, limitations, and potential customisations is essential for understanding how it can be tailored to specific needs, leading up to the formulation of a guide. Except for the first task, the rest of the tasks from this group will require having the Groundwork done.

- **OE1 -** Set up and deploy OpenWISP in a test environment: explore its modules, features, and potential areas of customisation. This task is planned to take 40 hours.
- **OE2** - Tailoring OpenWISP for Specific Needs: Attempt to tailor OpenWISP's functionalities to align with the needs identified during the groundwork phase. This task will take 25 hours. It can be done parallel to the previous task but as mentioned,

it requires the groundwork tasks completed, so that the tailoring makes sense. This task is planned to take 30 hours.

- **OE3** - Document challenges, solutions, and limitations found at OE1 and OE2, so it will be parallel to the previous task of this same group. It is planned to take 5 hours.
- **OE4** - Loop of Iterative Refinement: Include a regular refinement of the guide to ensure that new information is integrated seamlessly. This task it is planned to take 5 hours and can be parallel to OE3.

As the planning phase progresses, a recurring theme emerges: the technical intricacy of OpenWISP. While OpenWISP offers a powerful set of tools, there's an underlying concern about the technical expertise required to effectively operate it, especially within the context of Hahatay. Acknowledging this, it becomes apparent that the culmination of our groundwork and exploration tasks can be channelled into creating a comprehensive resource. By doing so, we can bridge the knowledge gap and assist users in navigating OpenWISP's complexities. This realisation leads to the **Guide Formation** (GF) tasks.

- **GF1** - Conceptualizing the Guide: structure the guide based on the identified challenges, solutions, and tailored functionalities while ensuring relevance and feasibility. This task and its subtasks are planned to take 15 hours. It does not need the exploration tasks completed, since ideas on how to structure the guide and its contents are likely to come to mind during those tasks.
    - GF1.1 Decide the chapters, sections, and subsections of the guide focusing on the user experience.
    - GF1.2 Review its relevance against already existing guides and OpenWISP’s documentation.
- GF2 - Drafting the Guide: Begin drafting, including practical examples, diagrams, and any relevant code or configuration snippets. It is planned to take 45 hours.

- GF3 - Review and Refinement: Share the draft with peers, professionals, or the thesis director for feedback and make necessary adjustments. It is planned to take 20 hours.
- GF4 - Finalizing and Publishing (5 hours): Prepare the guide for publishing, either online or as an Annex to the thesis. It is planned to take 5 hours.

Given the deep engagement with OpenWISP throughout the project, an inevitable outcome is the identification of areas for potential enhancement or modification. A task focused on an **OpenWISP Enhancement Proposal** (OEP) is in charge of formalising these insights into structured proposals.

- OEP1 - Idea Generation: Drawing from experiences, feedback, and iterative refinements, identify potential areas within OpenWISP that can benefit from improvements, modifications, or entirely new features. This is planned to take 5 hours.
- OEP2 - Technical Assessment: Evaluate the feasibility of these ideas. Understand their implications, dependencies, and potential impacts on the existing architecture and functionalities of OpenWISP. This task is planned to take approximately 5 hours.
- OEP3 - Proposal Drafting: Formulate a comprehensive proposal detailing the idea. This should include the problem statement, the proposed solution, benefits, potential challenges, and a high-level design or approach for its implementation. The estimated time for this is 15 hours.

The tasks of **Conclusion** (C) will remain the same, but the estimation of the **Documentation** (D) task increased by 10 hours. This is due to the fact that writing a guide involves a greater deal of revision because of the amount of text.

3.3 Initial budget

In this section, the economic impact of the project is analysed and discussed. Firstly the costs are identified as whether they are related to personnel, generic or indirect. We are going to refer to the tasks in the task definition section.

3.1.1 Personnel costs

To identify the human resources cost, it is mandatory to define different roles for the people going to take part in this project. For each worker, we will multiply the salary per hour by the time involved in the different activities.

We can recognise 3 types of personnel with different salaries: the Project Manager, Researcher and Technical Writer. All are performed by the researcher Alicia Pallarol Isábal, the Project Manager role, however, will be played by the GEP tutor and Thesis Director.

- Project Manager (PM): has the responsibility to plan, execute and monitor the progress of the thesis. It is summarised in the Project Planning (PP) tasks.
- Researcher (R): has the responsibility to do all the research work related to this thesis: Groundwork (GW), Experimentation (EX) and Conclusion (C) task groups.
- Technical Writer (TW): has the responsibility to document all the work done by the other two roles.

The different annual costs can be found in Table 2 and in Table 3 we can see the estimated hours for each role and the different task groups defined.

Role	Gross Annual Salary (€)	Gross Annual Salary + Social Security (€)	Hourly Salary (€)
Project Manager	41 052	55420.2	26.64
Researcher	32 847	44343.45	21.32
Technical Writer	38 884	52493.4	25.24

Table 2[2]: Annual and hourly salary for the different roles in this project. Wage with social security contribution calculated by multiplying by 1.35 the gross salary. Hourly salary is calculated by dividing the annual salary by the hours, days and months worked yearly assuming a 40-hour work week.

Name	Estimated Hours	PM	R	TW	Resources	Cost
Project planning	125	125	0	0	Laptop	3330
Groundwork	95	0	95	0	Laptop, Access Point, Ethernet cable	2025.4
Experimentation	255	0	255	0	Laptop	5436.6
Conclusions	15	0	15	0	Laptop	319.8
Documentation	65	0	0	65	Laptop	1640.6
Total	555	125	365	65		12752.4

Table 3[3]: Hour distribution for the different task groups and the responsible roles for each one. Using the hourly salaries in Table 2. The total hours exclude the hours needed for the oral defence.

3.1.2 Generic costs

To illustrate the project's economic cost, the amortisation of the resources employed needs to be included. Specifically, the hardware resources, since all the software utilised for the thesis is open source.

We will discuss the amortisation of the researcher's laptop and the hardware needed for experimentation. The laptop is a Lenovo ThinkPad T440p, it will be used for the complete 560 hours estimated for the project development. Specifically, we will be working 126 days an average of 6.3 hours a day for the 560 hours total of the defined tasks.

[2] Source: own elaboration according to the information at *Project Manager, Information Technology (IT)* (2023), *Technical Writer Salary in Spain* (2023) and *Research Scientist Salary in Spain* (2023).
[3] Source: own elaboration according to the initial task definition.

As for the rest of the Hardware needed, it will be provided by the AC department of the university and will be used in the Experimentation group of tasks, for one of the iterations of the EX1 and EX2. A complete iteration of the Experimentation group is 85 hours, but without EX3, those items will be used for 65 hours, since each iteration without EX3 lasts 35 days, the hours per day are 1.85. The access point price is 20€ and the Ethernet cable is 2.69€.

The formula for the amortisation is as follows:

$$Amortisation = Resource\ price * \frac{1}{Years\ of\ use} * \frac{1}{Days\ of\ work} * \frac{1}{Hours\ per\ day} * Hours\ used$$

In Table 4 we find a summary of the amortisation for the hardware.

Hardware	Resource price (€)	Years of use	Days of work	Hours per day	Hours used	Amortisation (€)
Lenovo Thinkpad T440p	1150	10	126	6.3	560	81.13
Access point	20	4	35	1.85	65	5.02
Ethernet cable	2.69	5	35	1.85	65	0.54

Table 4[4]: Summary of amortisation of hardware components required in this thesis.

There are other generic costs we need to take into account:

- Electric cost: according to Red Eléctrica de España (REE) (n.d.) the most expensive price for the kWh as of the 13th of March of 2023 is 0.127€/kWh. This price will be our reference for the calculation of electricity. We will only count the electric cost of the Laptop since the access point and the Ethernet cable consume so little energy (5W for the access point) and are used for a small amount of time that we are discussing a 0.05€ cost.

  As for the laptop, the average consumption is 100W since it is an old laptop, so for 560 hours, we will be using 560kWh and the cost for that will be 71.12€.

[4] Source: own elaboration.

- Workspace, Internet and water: since the researcher lives in a residence hall in Barcelona, the rent of the workspace is 560€ with furniture, Internet and water included and those prices cannot be segregated from the rent. Since this project is 5 months long, the total cost of the workspace, furniture, Internet and water is 2800€.

We do not have any cost related to software as we are going to use open-source software for all tasks from documentation to experimentation. Moreover, all meetings with the Thesis Director will be held through the online platform Jitsi Meet, which means there will not be any travel expenses.

In Table 5 we can find a summary of the direct and indirect costs:

Personnel costs (€):	**12752.4**
Generic costs	**Amount (€)**
Amortisation of Lenovo Laptop	81.13
Amortisation of Access Point	5.02
Amortisation of Ethernet cable	0.54
Electricity	71.12
Workspace	2800
Total personnel + generic costs (€):	**15710.21**

Table 5[5]: Personnel and generic costs.

3.1.3 Other costs

There are two very important costs to take into account:

- Contingencies: it is essential to be prepared for unforeseen circumstances, this accounts for the planning but also the budgeting. For this reason, it is reasonable to set up a fund to confront the different risks. Since the consequence of our main risks is having to adjust timing but at the same time all work will be based on a single and

[5] Source: own elaboration.

old laptop, both personnel and general costs need a proportional margin. From Table 4 we have that the combined cost of personnel and general is 15710.21€ so the margin for contingencies should be the 15% of that combination: 2356.54€.

- Incidental costs: according to our Risk Analysis in Table 6 we have a numerical evaluation of these obstacles with the impact in hours and in its cost (€) depending on the role that would need to work and the salary, the likelihood of that happening (risk) and also the cost that each possibility will be added to the budget accounting to the risk:

Possible Incident	Impact (hours)	Risk (%)	Estimated cost (€)	Cost accounting to risk (€)
Deadlines	30	25	725	181.25
Lack of experience	20	40	426.4	170.56
Poor software maintenance	15	10	319.8	31.98
Deficient documentation	5	10	126.2	12.62
Total (€):				**396.41**

Table 6[6]: Incidental budget calculation. The cost estimations for the incidents depending on roles are as follows: in deadlines, 5 hours will need to be dedicated by the Project Manager to reorganise, 10 hours by the Researcher and 15 hours by the Technical Writer; as for the lack of experience and poor software maintenance, all the hours are linked to the Researcher. Finally, for the deficient documentation, all 5 hours are accounted to the Technical Writer.

Table 7 summarises the other costs:

[6] Source: own elaboration.

Cost name	Amount (€)
Contingencies	2356.54
Incidents	396.41
Total (€):	**2752.95**

Table 7[7]: Summarization of contingencies and incidents.

Table 8 summarises the complete budget:

Cost name	Amount (€)
Personnel costs	12752.4
Generic Costs	2957.81
Other costs	2752.95
Total (€):	**18463.16**

Table 8[8]: Budget summary.

3.4. Planning and Timeline

The approximate duration of this project is 550 hours distributed in 35 weeks: from mid-February to mid-October. This gives almost 16 hours weekly dedicated to the project, which will imply a little above 3 hours a day.

3.4.1. Initial Schedule and Planning

The initial planning, in a Gantt diagram can be found in Figure 1. It follows the tasks described in the Initial tasks section.

[7] Source: own elaboration.
[8] Source: own elaboration.

Figure 1[9]: Gantt diagram with the initial planning.

3.4.2. Deviations and Adjustments

The planning in Figure 1 showed some tasks and times that do not correspond to the Final tasks. This is due to a few reasons.

1. Change of direction: while starting to look at OpenWISP, it was evident that a not in-depth analysis would not lead to findings or implementations. For this reason, it was decided to focus on this software platform.

[9] Source: own elaboration.

2. The opportunity to get involved in the Battle Mesh came along, and it seemed like a great learning opportunity that could lead to huge advancements in the understanding of the platform. However, while this deviation provided more perspectives and also contacts, it did take time away from the writing and structuring of the thesis.

Taking into account all of this, the decision to move the date from June to October was made.

3.4.3. Impact on Objectives and Costs

These changes impacted objectives as initially they were more general and included finding various solutions among different platforms. However, the learning outcomes have not changed.

About the cost, the amount of hours dedicated to each section changed, so it has been necessary to make some minor adjustments found in Table 9, Table 10 and Table 11:

Name	Estimated Hours	PM	R	TW	Resources	Cost (€)
Project planning	135	135	0	0	Laptop	3596.4
Groundwork	130	0	130	0	Laptop, Access Point, Ethernet cable	2771.6
OpenWISP Functionality Exploration	80	0	80	0	Laptop, Access Point, Ethernet cable	1705.6
Guide Formation	85	0	85	0	Laptop	1812.2
OpenWISP Enhancement Proposal	25	0	25	0	Laptop	533
Conclusions	15	0	15	0	Laptop	319.8
Documentation	80	0	0	80	Laptop	2019.2
Total	545	135	330	80		12757.8

Table 9[10]: Hour distribution for the different task groups and the responsible roles for each one according to the final planning. Using the hourly salaries in Table 2. The total hours exclude the hours needed for the oral defence. This is a correction from Table 3 with the new hour distribution.

Hardware	Resource price (€)	Years of use	Days of work	Hours per day	Hours used	Amortisation (€)
Lenovo Thinkpad T440p	1150	10	175	3.11	545	115.16
Access point	20	4	70	3	210	5
Ethernet cable	2.69	5	70	3	210	0.67

Table 10[11]: Summary of amortisation of hardware components required in this thesis according to the final planning. This is a correction from Table 4 with the new hour distribution.

Personnel costs (€):	**12757.8**
Generic costs	**Amount (€)**
Amortisation of Lenovo Laptop	115.16
Amortisation of Access Point	5
Amortisation of Ethernet cable	0.67
Electricity	7.12
Workspace	5040
Total personnel + generic costs (€):	**17925.75**

Table 11[12]: Personnel and generic costs according to the final planning. This is a correction from Table 5 with the new hour distribution.

[10] Source: own elaboration according to the final task definition.
[11] Source: own elaboration.
[12] Source: own elaboration.

These new updated tables follow the calculation rules, for the amortisation, for example, as in the initial budget. Also, the “other costs” described at section 3.1.3 will remain the same to compute the global budget since its concepts are Contingencies and Incidents for a total of 2752.95€. The electricity costs were miscalculated in the initial budget, so we need to correct them: for a 100W laptop running 545 hours we get 54.5 kWh and for the 5W access point 1.5kWh which is a total of

The updated total is 20678.70€ with the original budget at 18463.16€. So this project went 2215,54€ over budget, mainly due to the rent since there was a huge addition of months for the workspace.

Luckily, we had a margin of contingency of 2356.54€ so even though it was pretty tight, the cost overrun of the workspace fits into the total initial estimation.

3.4.4. Final Planning and Timeline

The new planning can be found in Figure 2 in the form of a Gantt diagram, scaled to weeks.

Figure 2[13]: Gantt diagram with the final planning.

[13] Source: own elaboration.

4 Sustainability report

It is essential that as a society we start integrating a sustainability dimension with all its implications (economic, social and environmental) in all professional processes: from academic to industrial. It is not enough to think of the economic aspect.

For this reason, it is crucial to analyse the sustainability impact of this thesis. From the economic implication that we already discussed a little bit with the amortisation, but also the social impact this project has, reflecting on how we can serve others and improve others' welfare; finally, we can not forget the consideration for the environment: we need to take into account the footprint of our project, how can we lessen the environmental impact.

In this report the sustainability is discussed following the sustainability matrix provided for the grading of the thesis and therefore containing a section on the Project Put into Production (PPP), the lifespan and the risks.

4.1. The Project Put into Production (PPP)

It is very interesting to have the PPP analysed under different perspectives:

- Environmental: It is very difficult to quantise the environmental impact of a thesis like this in terms different from the kWh. This is because this project uses little hardware and does not require big cloud storage solutions, for example. In *section 3.4.3* the budget was estimated taking into account the kWh and we had 545 hours of usage of a laptop, 210 hours of usage of an access point and Ethernet cable, however, the Ethernet cable consumption is negligible. For the laptop we are estimating 100W because it is an old computer and for the access point around 5W. For the laptop it would be 54.5kWh and for the access point 1.05kWh. Moreover, this thesis could

have been done with almost half of the time for the access point, which could have saved energy.

- Economic: For the budget there was a big time deviation that with the costs of the amortisation and rent, for example made a big difference. However, it fitted in the budget of the contingency plan. It is explained in *section 3.4.3.*
- Social: Personally, realising that this project even with the little amount of hardware is more than 50 times what a fridge consumes, is surprising. It is difficult to think about this since we use computers as tools for what we want to achieve, they are an extension of ourselves sometimes. Nonetheless, a platform such as OpenWISP proves to make networking infrastructures more efficient, however, this fact is not that good because it could cause a rebound effect implying that the infrastructures are bigger because of their improved efficiency. Balancing this information during the elaboration of the thesis has been rather overwhelming because as engineering students we are taught that efficiency is the goal, but when it comes to the real world, efficiency may just mean more consumption thanks to better resource management.

4.2. The Lifespan

In this section the discussion also focuses on the same parameters as the previous one:

- Environmental: The resources used on the lifespan are the same ones as for the PPP. As also discussed, the impact of the efficiency increase will be at a first glance good for the global footprint, since with the same resources the footprint will be less or even less resources will be needed. However, there is a personal concern on the rebound effect.
- Economic: The costs of the project on its lifespan are the same ones as for the PPP. Even the updates and adjustments have been researched and they do not imply the use of further resources.

- Social: Communities in need of OpenWISP tools can benefit from this thesis' work since it helps to bridge the knowledge gap required to operate OpenWISP. All kinds of organisations can get a grasp of the possibilities it has to offer so that they can deploy it. The content of the guide provides the information needed to know what direction to take when intending to deploy OpenWISP and this usage curve is at the very minimum flattened.

4.3. Risks

There are plenty of risks worth analysing from the same points of view we have been using::

- Environmental: for the project, a risk is the use of inefficient hardware. New computers, for example, consume less power. However, we could have also deployed the testing environments in real servers instead of a having virtual machines in a laptop, which clearly would have had a bigger footprint, so we had some balance.
- Economic: scenarios compromising the viability of the project include inadequate planning, lack of resources, lack of time, lack of expertise and technical challenges. All of those could have set back the project for more time that would have not fit in the contingency plan. To mitigate these risks it is important to have support so that it is possible to gain in planning skills. Planning involving allocating extra time taking into account the possible impact of the lack of expertise and also having a flexible methodology that allows the project to evolve according to the difficulties faced.
- Social: since OpenWISP is an open source platform, there is no risk involving a vendor lock-in. However, for network administrators, the risk could be to have a problem that no one in the community has been before or that the community does not really help you out. These scenarios could be risky for the project but could be mitigated through consistent participation and involvement with the community.

4.4. Conclusions

We are members of UPC, a public university located in a fully developed country, a welfare state; we have the responsibility to look at the globalised world and take into account that world issues, even if they are geographically far from home, are matters that involve us. They involve our consumption models, the way we interact with others and how we, as humans, want progress to look. Everything.

After writing this report, it is easier to look into future situations where it will be easier to recognise the best approach in terms of environmental, economic and social sustainability. But it is necessary that we develop these realisations earlier in life, in order to start making the best possible solutions quickly.

Finally, sustainability, as a concept, could be more integrated in the contents of all degrees, specially technical degrees because it makes professionals get in touch with the society from other points of view other than just technology and having the most amount possible of points of view helps us to understand others, the world and even ourselves better.

5 OpenWISP and Organisations: Customisation and Network Optimisation

The first thing we need to evaluate the relevance of OpenWISP for organisations is an analysis on what the different features network infrastructures in most organisations need. In this section some of the most important features will be discussed, specifically the features related to customisation and optimisation, because the foundation of a robust and resilient network not lies just in the hardware.

The customisation aspect ensures that a network aligns with the unique needs of its users and administrators, catering to specific user roles, policy enforcement, and any necessary automation. Optimisation, on the other hand, is about enhancing the network's efficiency, ensuring smooth traffic flow, and selecting the best channels for communication.

OpenWISP as a modular platform, offers great detail as per customisation and optimisation so in this section, we will be discussing the user roles, policies, channel selection and best practices in general.

One of OpenWISP's primary features is its capability to manage multiple organisations. With Hahatay in mind, we will only be using one, so in the following sections we will be discussing within an organisation, assuming we only have one, the most important topics to keep in mind when discussing network optimisation and customisation and their relation to OpenWISP.

5.1. Specific needs of Hahatay

The most remarkable network infrastructure related needs Hahatay has are related to the user management. This is because at the starting point of this project, it was possible that while in a class, for example, someone passing by could use up the bandwidth not allowing the class to carry on properly.

Looking into this, the first idea was to look up for how to implement VLANs, however, once seeing that OpenWISP has RADIUS support it was really worthy to explore its potential.

Moreover, at Hahatay, coverage is also an issue since the buildings are built with mud bricks and the material seems to complicate the signal transmission and reception, which is a topic interesting to investigate further.

As any other network, in Hahatay having an efficient and secure network are also requirements and all mentioned matters are discussed in the upcoming sections.

5.2. Customisation: User Roles and Network Policies

The first thing in discussion when talking about customisation is the relevance of user roles and policies. Distinct user roles ensure that individuals have access to the resources they require without compromising the network's security. Policies, meanwhile, set the stage for consistent and streamlined network operations.

5.2.1. User roles

The implementation of user roles, often via mechanisms like Role-Based Access Control (RBAC)(Fortinet, n.d.). RBAC essentially is a system of arranging access to network

resources based on the roles of individual users within an organisation, and it offers several significant advantages. Firstly, it bolsters security measures by ensuring that potential threats from human errors are limited.

Consider a scenario where an employee in marketing is compromised through a phishing attack; the risk is significantly mitigated if their role does not grant them access to sensitive company data.

Furthermore, RBAC simplifies the task for network administrators. Instead of individually adjusting access for every user, they can modify permissions based on roles, leading to a streamlined process that ensures users get timely access to the network resources necessary for their tasks.

Since having roles is so significant, our first contact with OpenWISP functionalities will be related to this. The platform distinguishes between two primary user contexts: 'network administration', which involves individuals responsible for setting up, monitoring, and maintaining the network, and 'end-users', who primarily connect to and use the network.
The network administration as a user context is related to the permissions each individual can have according to their role in the network administration. The end-users, on the other hand, go hand-in-hand with OpenWISP's integration with RADIUS, a networking protocol, to manage the authentication, authorization, and accounting for these end-users. RADIUS makes it easier to manage user access to the network.

All these advantages would be very beneficial for Hahatay: imagine a scenario where a volunteer needs access to specific network resources for an event. With RBAC, their access

can be easily managed and restricted once the event concludes. And for the end-users, it could help to segment the network, so there are no usage abuses.

Let's dig deeper now into the network administration users: it is possible to have different roles, as groups and then assign those groups to specific people, for example:

- Administrator: this role has a high level of access, it has a broader oversight and organisational structuring. It can add users, groups, all kinds of configurations, etc. Essentially, manages the network from the organisation side rather than the technical one.
- Technician/operator: this role interacts more frequently with the system, its permissions are related to operational tasks. For this reason, it has a great deal of access.
- Auditor: this role can view settings and logs, but not make changes. It can be useful for evaluations of the network.

The creation of each role or adapting them to different policies is a pretty straightforward process in OpenWISP. No roles are predefined aside from the Administrator and the Operator, all the permissions can be changed for all roles.

5.2.2. Authentication and Access

The end-users can be created in batches, from a .csv file, for example, and there is also the possibility to create groups for different users. The groups can have different custom "Checks" as they mention in OpenWISP where different aspects of the connection can be limited, for example, the daily traffic or the daily surfing time.

The actual connection for the end-users, using RADIUS, is done with a captive portal package named Coova-Chilli, this package is the one that does the Accounting, Authorisation and Authentication (OpenWISP, n.d-e).

The users can self-register using their mail or social accounts. Furthermore, paid plans can be implemented and be shown at the self-registration to have different limits to their connections. These payments are done via PayPal.

In the case of Hahatay, it may be helpful to manage the day-to-day users at the highest level. However, it could also be helpful to have a self-registration option designed just for visits and, parallelly, the recurrent users are created externally so that it is easier to manage in terms of the groups they belong to and the limits.

So a day-to-day user could self-register and use this account temporarily while his or her long-term account is being created by the network operator, for example. Or a new network administration role could be created with user management permissions just to manage new registrations.

Some examples of groups of users that could be created are: staff, students, teachers and guests. Each one could and should have different limits.

5.2.3. Monitoring and Data Collection:

OpenWISP can be configured to track user activity logs. If that were the case, it would record the following information for each user: username, IP address, MAC address, connection start time, connection end time, data usage, and visited URLs.

This data can be used to monitor user activity and identify potential security threats. For example, if a user is spending a lot of time on a particular website, it might be good to investigate further to see if they are visiting a malicious site.

Additionally, it can be used to comply with regulations: OpenWISP can be used to collect the data that is needed to comply with legal regulations.

5.2.4. Network Policies

At Hahatay, as in any network, it is important to define on paper, with proper documentation, the definition of network administration roles and permissions, end-user roles as well as the justification behind the collected data.

This is important because there can be changes in the technical staff. Having well-documented policies ensures that the new staff member can quickly understand the network's setup speeding up the training and onboarding process, allowing them to get up-to-speed faster and start managing the network faster.

Moreover, over time, as networks evolve and grow, decisions are made for specific reasons—maybe to address certain challenges or requirements. Documenting these reasons can provide context for future decisions.

It can also help to standardise processes: clear documentation can help in standardizing certain processes, ensuring that best practices are always followed. Also, if any issues arise or if there's a need for audits, having clear documentation can provide clarity on what was done, when, and why.

Finally, as technology and user needs evolve, periodically reviewing the documented policies can help identify areas for improvement or changes.

5.3. Optimisation: Channel Selection, Traffic Management, and VLAN Implementation

After discussing the policies and user roles, it is time to discuss the optimisations possible with OpenWISP.

5.3.1. Channel Selection

The channel used is an important item in terms of performance. For this reason, the channel selection should be discussed as a means of improving the Quality of Service (QoS). QoS are all the tools that according to Fortinet (2023), “control traffic and ensure the performance of critical applications”.

The selection of the channel should be a mindful one, taking into account the suitability for different environments (outside/inside) and the possible interferences with other networks.

In Hahatay this topic is rather relevant since, as we mentioned, the buildings are built out of mud bricks. Currently, this is causing issues related to interference and signal degradation so it is important to keep the channels in the 2.4GHz band. Low-frequency waves transmit better through a wall according to Intel (n.d.).

As the most used channels for Wi-Fi are the 2.4GHz we should also be mindful about the possibility of their overlapping. For this reason, in dense areas, it would be good to make sure that we are using the non-overlapping 1, 6 and 11 channels (Cisco Meraki, 2020).

In OpenWISP automatic channel selection, from version 2.0.0 does not exist. For this reason, the only way to change the channel is to manually change the values of the channel at each device. Since there is no possibility to enable automatic selection, a Wi-Fi analyser tool could be used to find out the state of the network to select the most appropriate channels at any given time. For example, the famous Wireshark can be a Wi-Fi analyser (Wireshark, n.d.). It can be installed on any device connected to the network to evaluate this, for example.

5.3.2. Traffic Management

Given the distinct user roles, it is imperative to balance user demands with available bandwidth. Even with each user having a defined role, it is important to delve further into Quality of Service (QoS).

We mention the possibility of limiting network usage, and in terms of traffic management, setting up policies with those limits comes with pros and cons. Pros:

- Reduced congestion: It can help to reduce congestion and improve performance.
- Fairness: It can help to ensure that all users have a fair share of the bandwidth.
- Cost savings: It can help to reduce the cost of the network.

Cons:

- Reduced user experience: It can negatively impact the user experience.
- Increased administration: It can require additional administration.

OpenWISP offers some modules that can assist in understanding network traffic patterns, which can further inform optimisation strategies. The monitoring module includes traffic and bandwidth usage charts. We will provide more details on what the monitoring module offers in *chapter 5*.

There is a third possibility to enforce QoS with OpenWISP by using 3rd party software such as Quagga (Quagga, n.d.). However, we are not going to delve deeper into them due to the issues that 3rd party services bring to the table with QoS, some of the issues are:

- Complexity: The configuration is highly challenging and there is no straightforward solution or straightforward integration.
- Cost and vendor lock-in: They can be expensive and if they are not, the platform may lock the deployment into that vendor's platform. This can make it difficult to switch to a different QoS solution in the future.
- Performance: They can add latency and overhead to the network traffic, which can be a problem for applications that require low latency, such as Voice over IP (VoIP) and video streaming.
- Security: They can introduce security risks to the network because these services often have access to sensitive network traffic data.

5.3.3. VLAN Implementation

According to IBM (2023) a VLAN, which is a logical broadcast domain, splits up groups of network users on a real physical network onto segments of logical networks.

It has plenty of benefits, let's highlight some (University of Wollongong, 2008):

- Ease of administration: VLANs focus on devices separated in the network. For this reason, a change of location does not imply the need to reconfigure the end stations.
- Confinement of broadcast domains: With VLANs the need for deployed routers on the network containing broadcast traffic is significantly reduced since the flooding of a packet is restricted to the interfaces in the same VLAN. Actually, this confinement significantly reduces traffic. In some cases this traffic can arrive unwillingly, however, with a VLAN this traffic is avoided.

VLANs enhance both security and management efficiency. By segmenting the network, VLANs allow for more organised oversight, making them highly powerful for settings like Hahatay where distinct groups like faculty, students, administration and guests may require differentiated network access. Specifically, in Hahatay's context, VLANs could be instrumental in allocating higher bandwidth to classrooms during active sessions, ensuring uninterrupted lessons.

OpenWISP cannot currently configure VLANs within the platform. However, there are a few ways to work around this. One way is to use a third-party VLAN management tool. There are a number of these tools available, and they can be used to configure VLANs on a variety of devices. The general way to integrate a 3rd party service for VLANs in OpenWISP would be by using the netjsonconfig protocol. The netjsonconfig is according to OpenWISP (n.d. -b) “a python library that converts NetJSON DeviceConfiguration objects into real router configurations that can be installed on systems like OpenWRT, LEDE or OpenWisp Firmware”.

The third-party VLAN management tool would need to be able to send and receive netjsonconfig messages. Once the third-party VLAN management tool can communicate with the OpenWISP controller, it will be able to use the netjsonconfig protocol to manage VLANs on the OpenWrt devices that are managed by OpenWISP. OpenWRT is a Linux-based firmware used in devices such as routers, access points, for example (OpenWRT, 2023b).

However, given that all of Hahatay's routers and network devices operate on an OpenWrt backend, VLANs can be directly configured by accessing the individual OpenWrt interfaces on these devices. The process is straightforward, primarily requiring uci (Unified Configuration Interface) commands to add the VLAN to the devices and then assign the VLAN to a switch port. According to OpenWRT (2023a) uci “is a system to centralize the

configuration of OpenWrt services". To do this, it is required to edit the switch port configuration which is different depending on the model.

5.4. Best Practices and Challenges

Building a culture rooted in best practices is essential, especially in complex infrastructures like Hahatay. The following best practices can enhance network performance and security:

- Regular Updates: Always keep OpenWISP and any other integrated systems up-to-date to benefit from the latest security patches and features.
- Backup and Recovery: Establish a routine for backing up configurations, user data, and essential settings. It is also essential to discuss the importance of having a disaster recovery plan.
- Monitoring and Alerts: Set up monitoring tools to keep an eye on network health, usage patterns, and system metrics. Utilize alert mechanisms to get informed about potential issues before they escalate.
- User Training: Ensure that users -especially those in administrative or technical roles- receive adequate training on the system. This not only empowers them but also reduces the risk of misconfigurations due to human error.
- Documentation: Maintain comprehensive documentation on the network setup, configurations, and any custom solutions implemented. This assists in troubleshooting and is invaluable during handovers or staff transitions.

There are also plenty of security best practices:

- Password Policies: Emphasize the importance of strong, unique passwords. Consider periodic password changes.
- Implementing correctly the Role-Based Access Control (RBAC): Ensure users have only the access they absolutely need.

- Regular Audits: Periodically review user roles, privileges, and system access to ensure no inadvertent security lapses.
- Firewalls and Intrusion Detection: Implement and regularly update firewall rules. Consider intrusion detection systems to detect unauthorized access attempts.
- Physical Security: While much of the discussion might be about digital security, it is crucial to mention the importance of securing the physical hardware and access points.

There are also a set of challenges linked to the setting up of an infrastructure:

- Scalability: As the network grows or user demands increase, there might be challenges in scaling up the infrastructure or managing a larger user base.
- Interoperability: If integrating with other systems or platforms, like the mentioned 3rd parties, there might be challenges in ensuring seamless operation.
- Technical expertise: While OpenWISP is designed to be user-friendly, certain advanced features or troubleshooting might require a deeper technical understanding. An operator of the network must be consistently reachable for issues that may come up.
- External threat landscape: The external threat environment is dynamic, with new vulnerabilities and challenges emerging regularly. So staying updated and proactive is a constant challenge.
- User resistance: Especially in established environments, there might be resistance from users in adapting to new systems or policies.

6 Ubiquitous Access and User-Centred Design

A consistent and comprehensive network is necessary in bridging the digital divide and fostering digital inclusivity. It forms the backbone for seamless learning and enhanced educational opportunities.

Take, for instance, online tests. In schools, where resources are scarce, uninterrupted online tests become a lifeline for assessment. A reliable network ensures that students can complete their tests without disruptions, thereby enabling fair evaluation and reducing educational disparities.

Moreover, a consistent network allows teachers to access educational materials, conduct research, and employ multimedia tools effectively. This enriches the learning experience, making it more engaging and informative for students. For instance, a teacher in a remote village can bring global knowledge into their classroom, broadening horizons that might otherwise remain constrained.

The fact is that reliable networks facilitate administrative tasks. The efficiency increase can be transformative, freeing up resources for educational improvements.

All in all, not only in schools but in all kinds of organisations, a consistent and comprehensive network transcends being just a technological necessity. It becomes the bridge that connects people to a world of opportunities.

6.1. Ensuring Comprehensive Network Coverage

In organisations with many facilities or huge ones, there can be many areas to cover by a network. Hahatay, by being not only a school but a community, also has many facilities and areas, for instance, a cultural house, a women's centre and a school.

These different facilities will have different sets of walls, doors, furniture, etc. For this reason, a Site Survey is needed. A Site Survey, according to Cuevas (2020) is a comprehensive evaluation of wireless networks and their relationship with the physical environment of a network. Their purpose is to gather precise information in order to determine the number and placement of access points required to ensure efficient and uniform coverage throughout the space.

For example, they identify potential sources of interference that could negatively impact the performance of the wireless network, like microwaves or other electronic devices. Radio frequency signals can be influenced by various obstacles, such as different types of objects and materials, which can disrupt the signals and cause reduced coverage.

So by conducting a Site Survey, it is possible to gain a detailed understanding of how wireless signals behave and propagate in their environment, enabling to optimise the wireless network setup for optimal performance.

Since there are many spaces to be covered by the network, Hahatay already has in place a mesh network to cover the most area possible. A mesh network is a Wi-Fi system that prevents dead zones and provides uninterrupted Wi-Fi (TP-Link, n.d.). The main difference with traditional routers is that in mesh networking, there are multiple access points instead of broadcasting Wi-Fi from just one.

In heterogeneous scenarios, from University Campuses to military settings, mesh networks have proven their efficiency. When one unit connects to the modem, that unit becomes the main hub, while the other units, the nodes, capture and relay the signal from the router. This node-to-node relay system facilitates redundancy and adaptability, ensuring robust connectivity, even in terrains with variability. Data can navigate alternative paths around obstructions or interference, thanks to this architecture.

Furthermore, the self-healing nature of mesh networks allows nodes to dynamically adjust routing, maintaining uninterrupted connections. The result is a scalable and adaptable wireless network that conquers connectivity challenges in diverse physical landscapes, providing a strong and reliable signal.

Although you can not directly set up mesh networks with OpenWISP -it is required to configure the mesh from the OpenWrt system- its controller can connect with the mesh to centrally manage all the devices in the mesh network. This includes things like configuring SSIDs, assigning users, and monitoring network performance.

Moreover, the OpenWISP controller can be used to automatically update the firmware on all the devices in the mesh network. This ensures that all the devices are running the latest security patches and bug fixes. Additionally, it can be used to troubleshoot issues since the controller can provide information about the status of the network, such as which devices are connected and how much traffic is being generated.

As mentioned, in Hahatay they are already using all devices with OpenWrt, which is the only requirement to run OpenWISP.

Finally, it is essential to discuss scalability. To achieve it, in any circumstance, it is necessary to build a flexible network infrastructure. This means investing in hardware and software solutions that can adapt to growing demands. For example, opting for cloud-based services can provide scalability on demand, reducing the need for frequent and costly hardware upgrades.

Periodic network audits are another crucial element in maintaining an optimal network. These audits should encompass technical assessments as well as a thorough examination of how the network aligns with changing educational requirements. By regularly assessing performance and identifying potential bottlenecks or areas that require improvement, organisations can proactively address issues before they become impediments to learning.

Feedback from the community at Hahatay is invaluable. Their insights can shed light on user experiences and pinpoint challenges that may not be apparent through technical assessments alone. Encouraging open communication and actively soliciting feedback ensures that the network is aligned with the real-world needs of its users.

Budgeting for network upgrades is also critical. For this we need to recognise that technology evolves rapidly, and allocating resources for updates and improvements is essential to keep the network current and capable of supporting educational goals effectively.

Lastly, investing in training for all members of the community is vital. With new technologies and tools continually emerging, ensuring that the community has the knowledge and skills to make the most out of these resources optimally is key to the long-term success of the network and, ultimately, the quality of education and other services provided.

6.2. Design of the User Guide

One of this thesis goals is the compilation of a guide that complements OpenWISP documentation, the guide can be found in the Appendix A. It was conceived as a digital guide, because although it follows an academic format with references, it also contains links within the text that are not referenced because they lead to OpenWISP's documentation on specific topics.

In this section the design principles followed to write the guide are explained. Starting with the fact that user-friendly documentation is indispensable for any technological system, especially in self-managed environments like Hahatay. It bridges knowledge gaps, enhances usability, and empowers diverse users. Clear guides streamline support, reducing disruptions to the educational process.

The goal for a user guide taking the example of Hahatay was that not only specialised networking professionals in Hahatay's community were able to deploy OpenWISP, but that other technicians can also understand its potential and be a part of it.

Our assessment of existing OpenWISP documentation revealed several challenges, including:

- The fact that the architecture is just explained in a photograph. It is difficult to understand how the different components work with each other.
- Since there have been different versions and it is still under development, the releases of new versions complicate the documentation, which in many cases is not updated.
- Without the proper understanding of the architecture, how the modules work and interact is not trivial nor straightforward.

- Even though the Dashboard on the graphical interface is straightforward, the possibilities OpenWISP offers for managing networks in organisations' infrastructures are not clear.
- There is a mailing list, but there is no place centralised that contains all the documentation for the modules and functionalities. So users are dependable on the answers of the community.

6.2.1. Outline of the guide

Before outlining the guide, we should emphasise that we tried to make the guide inclusive, including the various important topics relevant for all kinds of organisations as described in *chapter 5* and also supporting users to troubleshoot common issues themselves, fostering self-reliance.

Moreover, if the end goal of the guide is to be usable, it should have frequent reviews, in order to adapt to updates. As technology evolves and the needs of Hahatay's community change, the user guide must remain updated. Regular, continual revisions are not just best practices but crucial to ensuring that the guide's content aligns with real-world requirements and remains user-friendly. By maintaining the guide's currency, we ensure its longevity and relevance.

The methodology for navigating all the requirements included meetings with stakeholders at Hahatay and iterative revisions to actually make it user-friendly. The iterative revisions, involved fellow students with a deep networking knowledge but unfamiliar with the guide's content, ensuring a fresh perspective that helps identify potential issues.

To achieve all the described the outline of the guide includes these main chapters:

A. OpenWISP architecture: A chapter discussing the architecture of OpenWISP.
B. Customisation: A chapter discussing the topics covered in *chapter 5* in relation to OpenWISP's capabilities and adaptability.
C. Network optimisation: A chapter discussing optimisations that are allowed with OpenWISP.
D. User design: A chapter discussing ways to make the network usage and management as user-friendly as possible.
E. Case study: A chapter discussing how to evaluate an OpenWISP set up.

6.2.2. Testing environment to familiarise with OpenWISP:

In order to elaborate the guide a test environment was created. The aim of this experimentation was to simulate a small network infrastructure to assess the complexity of the whole process and to realise the features worthy of attention to the guide.

To do so, we used a virtual machine with Debian 11 as the operating system to set up the server and at the beginning we connected to it virtual machines running OpenWRT as devices, on a last instance we were also able to connect a router. The tutorials and documentation were always followed with the focus on simulating a production environment, for this reason we set up the server with Ansible. Ansible is a tool to automate different configurations and orchestrate workflows, among others (Ansible, 2019).

We were able to explore the different items of the OpenWISP Dashboard and see all the default values and possibilities. However, we were not able to apply configurations on the devices.

With our test environment the content compilation for the guide is more meaningful and directly addresses what a network administrator wanting to deploy OpenWISP for her or his own infrastructure would come across. The outcomes of this environment testing are further discussed in *section 7.1*.

6.2.3. Other considerations on the guide

To add relevance to the guide for organisations that may use it, targeted trainings, such as workshops for specific roles like operators or administrators, should be offered. These sessions, open to all interested community members, can foster understanding and confidence in using the technology.

7 Evaluations, Iterative Improvements and Proposal

The journey of improving and refining technological systems, like OpenWISP, is a continuous cycle of evaluation, adaptation, and innovation. In this section, we will explore the measures taken to understand OpenWISP's strengths, areas of improvement, and the efforts to optimise it for broader applications. Through simulation, a cost-benefit analysis, and observing its adaptability over time, we aim to build a comprehensive understanding of the system's evolution. These evaluations not only facilitate enhancements in OpenWISP but also provide valuable lessons for similar platforms in the realm of community networks, especially the one at Hahatay.

7.1. Preliminary Evaluations and Lessons

7.1.1. Objectives of Simulations

The objectives of simulations described at *section 6.2.2* were to experiment with OpenWISP's functionalities to know how they would fit into a real deployment. Specifically, the functionalities that could improve Hahatay's infrastructure. Also, the hope was to understand better OpenWISP's architecture from the experimentation.

7.1.2. Overview of Results

For overviewing the results, it is needed to note the intention of trying a production environment using virtual machines (VMs) posed some inherent limitations. VMs, sharing resources with the host machine, might not mirror the performance of real-world deployments. Additionally, nuances in virtual network interfaces and hardware emulation can

introduce discrepancies from actual setups. This isolation of VMs can also sometimes prevent seamless communication with other systems.

The qualitative highlights are:

- Tutorial: The basic set-up of the server is a process with a curated tutorial at the GitHub repository.
- Documentation: There is no straightforward documentation on how to integrate the different modules, the controller and the API.
- Communication in Virtual Environments: Setting up communication between virtual machines and ensuring they successfully connect to each other can be a nuanced task. In our simulations, initial challenges were encountered due to network configurations within the virtual environment. Once addressed, we were able to successfully establish communication. This underscores the importance of detailed setup and configuration, especially when virtual environments are used to replicate real-world scenarios.
- Security Concerns: When setting up and experimenting with network systems, security can often become an unforeseen challenge. For instance, ensuring that communication between different VMs is encrypted or securing the OpenWISP server from potential breaches.
- User Experience: The dashboard of the OpenWISP controller was deeply explored with during the simulations. In general, the different tabs were clear: Home, Devices, Configuration, Users & Organizations, Geographic Info, CAS & Certificates, RADIUS, Monitoring, IPAM, Firmware, Network Topology, Subscriptions and Help. When looking at the general picture, the dashboard is very intuitive. However, when entering into detailed specification or configuration of, for example, a particular device, there are many checkboxes without context on what they are exactly. If there are

checkboxes, usually there also exists an “Advanced Mode (JSON)” where you can add the configuration parameters with the values. However, there is little documentation and very hard to find the different parameters.

7.1.3. Lessons Learned

There have been many lessons learned on the OpenWISP simulations:

- Ansible, a popular automation tool used for configuration management and application deployment, plays a very important role. Once Ansible’s dynamic and how it interacts with the modules are figured out, it makes everything very smooth.
- While using VMs for the OpenWISP simulations, one challenge encountered was related to Network Address Translation (NAT) configurations, particularly when trying to establish concurrent Secure Shell (SSH) sessions to different VMs. NAT is a method used in networking that allows a device to modify IP address information in packet headers while in transit, enabling one IP address to represent multiple internal IP addresses. The VMs, operating behind the NAT, required unique port forwarding rules to ensure successful incoming SSH connections. Misconfigurations or overlaps in these port assignments lead to connectivity challenges, highlighting the importance of meticulous network setup in virtual environments.
- OpenWISP has a lot of potential since it offers a wide range of features, moreover it is still under development, which means that further features could be included in the future.
- Rigour is key when working with projects under development, especially since documentation can be outdated, incomplete, or might not address all potential challenges. In such evolving projects, staying updated with the latest changes, actively participating in community discussions, and maintaining thorough records of one’s own processes become crucial to avoid pitfalls and redundancy.

- A good level of understanding of web architecture is needed to figure OpenWISP out.
- Value of Real-World Testing: Simulations offer valuable insights, but they cannot fully replicate the intricacies of real-world deployments. Hence, there's an inherent value in pilot tests or small-scale real-world deployments.

7.2. Cost and Benefit Analysis

Having gained insights into OpenWISP and its functionalities, let’s discuss the potential costs and benefits of its implementation within Hahatay's framework.

The hardware requirements for OpenWISP are relatively modest. Actually, it could be run on a variety of hardware platforms, including Raspberry Pis, laptops, and servers. Furthermore, the main requirement for running OpenWISP is that the devices run OpenWrt, which all the devices in Hahatay already do.

The number of manpower hours required to set up OpenWISP, on the other hand, is a matter more difficult to break down. On an already functioning network, such as Hahatay, the hours will depend on several factors, including the number of Access Points (APs), the complexity of the network and the level of experience with OpenWISP.

We need to take into account that the network at Hahatay is significantly complex, having 4 different areas growing into 5, as seen in Figure 3. The areas are interconnected with antennas. One of the areas has the router that is connected to the Internet. Each area has from the antenna that receives the signal, a mesh network around it providing Internet to all the facilities in said area.

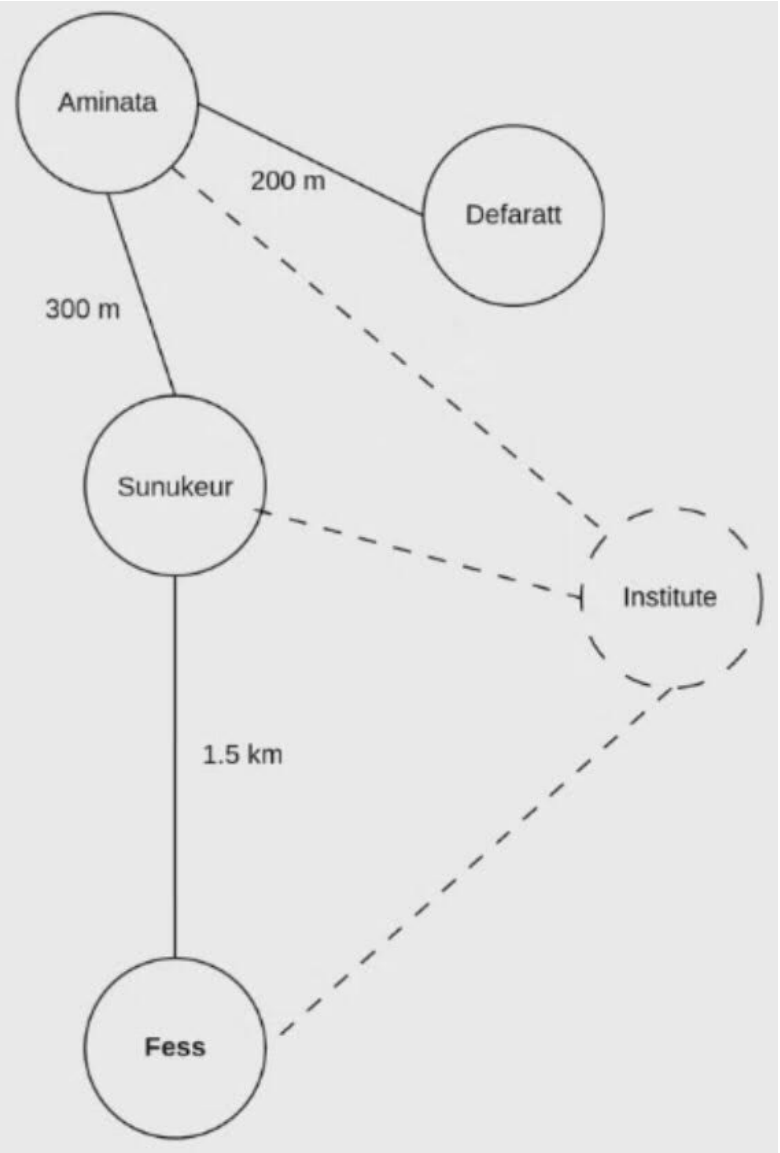

Figure 3[14]: Graph representing Hahatay's areas. The vertexes indicate the areas and the edges contain tags with the distance. The dashed lines refer to the fact that the Institute area is still under construction.

We can calculate that at the beginning, setting up an AP at Hahatay, where, as mentioned, all devices have already OpenWrt, can take up to 2.5 hours. This is because apart from installing OpenWISP on each OpenWrt device, it is required to configure the device to communicate with the OpenWISP server. This involves setting up the wireless network,

[14] Source: screenshot from a Hahatay

Dynamic Host Configuration Protocol (DHCP), and Domain Name System (DNS) and also testing it.

Moreover, as first steps, we find the initial set-up for the server, the end-user accounts, the network administration accounts and the generation of documentation. All of them imply a great number of hours. From the experience of the thesis, it is quite difficult to estimate, but with a clear hierarchy for the permissions, it should not take more than a week.

There are also costs associated with training the technical team to use OpenWISP effectively, especially if the system has nuances that the team is unfamiliar with. For the technical team deploying it, the guide will be a good resource that can save time and therefore money. However, hands-on workshops may as well be necessary to guarantee self-sustainability. The resources for organising these trainings should also be kept in mind.

Additionally, there are maintenance costs, since it is crucial to ensure that OpenWISP remains functional, updated, and secure.

Other costs worth discussing would be, for example, the opportunity cost. This is hardly the situation where investing time in OpenWISP implies sidelining other opportunities or projects. But a cost there may be, is related to the integration overhead because at the moment they have an infrastructure that with its flaws, works, so if challenges come up when integrating, they could cost money and imply a lack of connection for end users.

From using OpenWISP there are plenty of benefits and in this section, we will discuss, for example, the flexibility. This is because open-source solutions often offer more flexibility for customisation and integration. This allows the network to be tailored more closely to specific needs.

But there are more tangible benefits:

- Enhanced Network Management: OpenWISP provides centralised management and configuration of network devices. This can lead to reduced downtimes, faster issue resolution, and more streamlined operations.
- Direct Financial Savings: Using an open-source solution often means avoiding licensing fees. Moreover, enhanced efficiencies might lead to fewer resource requirements.
- Scalability: OpenWISP makes it easier to scale the network in the future, potentially reducing future capital expenditure.

OpenWISP represents a strategic investment not just in monetary terms but in future-proofing an organisation's network infrastructure. OpenWISP ensures that networks remain adaptive and robust. This foresight not only guarantees enhanced network capabilities but also positions communities like Hahatay for growth and innovation in the evolving digital landscape.

7.3. Adapting and Refining the Network Over Time

Static network designs can not accommodate the dynamism of networks. Needs evolve rapidly and so do the networks. It is mandatory to make networks as adaptable as possible. Just as expanding cities upgrade their infrastructure to accommodate growth, organisations must refine their networks. This ensures they remain robust, secure, and efficient.

As per monitoring the network, OpenWISP offers the OpenWISP monitoring module which offers according to openwisp (2023):

- Collection of monitoring data in a time-series database (influxdb) and essential device status metrics, including uptime, RAM, CPU load, and Wi-Fi status.

- Comprehensive monitoring charts covering packet loss, latency, and interface traffic, with varying resolutions from a day to a year.
- Configurable alerts for real-time network issue detection.
- Admin dashboard that offers a holistic view of network status, including device online/offline metrics and a geographic map (for OpenWISP users).
- Extensibility with custom metrics, charts, and an API for device data retrieval based on NetJSON DeviceMonitoring.

Other 3rd party solutions that can be integrated are Zabbix (Zabbix, n.d.) and Nagios (Nagios, n.d.), which are open-source. They do not offer very different metrics from OpenWISP but can be a good alternative if we are already familiar with them.

7.3.1. Key Performance Indicators (KPIs):

KPIs provide quantifiable metrics that offer insights into the efficiency, stability, and overall performance of a network. Regularly tracking these indicators ensures that the network meets its intended performance levels, aids in the early detection of potential issues, and can guide optimisation strategies. Let's delve into some pivotal KPIs that should be monitored within Hahatay's network infrastructure:

- Bandwidth Utilization: This metric helps in determining how much of the available bandwidth is actively being used to prevent bottlenecks.
- Network Uptime: Represents the percentage of time the network is operational without any disruptions in order to detect underlying issues.
- Packet Loss Rate: Represents the percentage of data packets sent from one network device to another that fail to arrive at their destination.
- Latency: Measures the time taken for data to travel from its source to its destination across the network. Lower latency ensures that applications and services function more responsively, improving user experience.

- Traffic Volume: By analysing the volume of data traffic on the network, one can determine peak usage times and potential sources of congestion.
- Number of Active Devices: Keeping a tab on the number of devices actively connected to the network can help in predicting load and ensuring that the network infrastructure can handle growth.
- Security Threat Detections: A metric that counts the number of detected security threats over time. An increase in this KPI could indicate a heightened risk environment or potential vulnerabilities in the network.
- Quality of Service (QoS): Evaluates the performance of specific data traffic types or applications. Ensuring that priority traffic (e.g., VoIP calls) receives the necessary resources is vital for consistent service delivery.

Monitoring these KPIs will not only provide a snapshot of the current health of the network but also guide future decisions related to upgrades, configurations, and expansions. By staying tuned to these indicators, Hahatay and any other network can ensure its network remains robust and ready to serve its evolving needs.

7.3.2. Evaluations and feedback

Beyond technical metrics, feedback from end-users provides invaluable insights into real-world performance and improvement areas. Regular assessments, whether quarterly or monthly, pinpoint network stress points or upgrade needs. A dedicated feedback channel, like a contact email, bridges the gap between end-users and operators.

7.3.3 Other considerations

Scalability ensures consistent service amidst growth or technological shifts. Tools like OpenWISP offer scalability, but third-party integrations require careful consideration to retain agility.

Additionally, given the diversity of users, regular security assessments and utilising tools like Nikto (Chris Sullo, 2021) could help, safeguard the network.

7.4. OpenWISP Enhancement Proposal (OEP)

During the Groundwork tasks and OpenWISP exploration, the opportunity to work closely for many hours with this platform has allowed for some reflections. Some reflections are mere technical observations, but some others are particularly interesting.

For example, it was a quest to find the documentation for the specific version chosen to work with. It was rather surprising the decentralisation of information, so the reflection on the possibility that this could be an aspect that might impede its wider adoption or efficient utilization, especially among novice developers is the cause of the following proposal.

7.4.1. Detailed Enhancement Proposal

This proposal is focused on the fact that there is a certain detachment of the community from the actual code. OpenWISP offers a mailing list as a means of communicating. Moreover, as an open-source project, there are Issues and Pull Requests opened and we can not miss the chance that there might be many local forks in developers computers.

This situation can be limiting for the not advanced developers who want to implement or customise certain modules. For this reason, one initial idea from the Literature Review came up again.

It was that centralising and standardizing the way of publishing different tweaks could be done by implementing a module marketplace where developers share their tweaks, so other

developers can find them all in the same place by searching keywords of what they need, for example. Also, it could intertwine the community further having GitHub repositories linked directly to the marketplace.

The marketplace could work as any app store for mobile phones does, with a name, a description and a logo. Modules would need reviews by community before being published as if they were a Pull Request to generate trust in the developers. Finally, it is a definitive condition that it must have a version checking, ensuring compatibility or at least discussing it verbosely in the description.

A more accurate description can be found at Appendix B where the Technical Design Document (TDD) describing all the features and implementation of the marketplace where the technical feasibility has been ensured.

7.4.2. Potential Impact and Importance

The potential benefits from this enhancement would be a better operator experience. It could imply a quicker response to new needs in a network.

Moreover, centralising and standardising the publication of tweaks and modules can streamline the process for developers. A dedicated module marketplace can act as a one-stop-shop, allowing developers to share, find, and even collaborate on custom tweaks and functionalities.

By streamlining the sharing and discovery of tweaks, this enhancement could facilitate quicker innovations, ensuring that the broader OpenWISP community can benefit from individual developer insights.

As an obstacle, the marketplace ideally should be a part of OpenWISP's page so it could be harder to integrate. Furthermore, implementing quality control could require additional resources or dedicated teams to review and verify the modules. Similarly, the open nature of a marketplace brings about security concerns that would require rigorous monitoring and swift response mechanisms. It is definitely not a trivial piece to add.

The OpenWISP community is a thriving ecosystem of developers and operators. By implementing a centralised module marketplace, we can foster an even tighter collaboration, streamline the customisation process, and propel OpenWISP to even greater heights.

8 Conclusion and Future Directions

This work comes from evaluating a very general situation focusing on the specific case of Hahatay while maintaining the universality of the solutions at maximum. Achieving a balance among it all has been a challenge. Through this balance, we have contributed to a finer understanding of network infrastructure optimisation, tailored not only to Hahatay's specific needs but also to provide insights universally applicable to similar scenarios in various organisational contexts.

8.1. Key Findings

Throughout the elaboration of this thesis we have come across different findings and reflections that could be summarized with:

- Dynamic nature of networks: Throughout this project, it became apparent that networks are not static. As organisations grow and evolve, network infrastructures need to adapt to cater to new operational demands and technological advancements.
- Importance of permissions and hierarchies: Ensuring a clear hierarchy for permissions not only streamlines processes but also significantly reinforces security measures, mitigating potential risks.
- Continuous monitoring and adaptation: With tools like OpenWISP and its monitoring module, maintaining an up-to-date view of the network's health is not just achievable but necessary. Regularly monitoring KPIs helps in identifying potential issues before they escalate.
- Community involvement in open source: OpenWISP's strength lies not just in its codebase but also in its active community. However, the detachment observed between the community and the core code has potential implications for its growth and accessibility.

- Potential of module marketplace: The idea of centralising customisations through a marketplace is not just about simplifying the developer experience but also about amplifying community collaboration. This centralisation is feasible if not as an integral part of the OpenWISP page, available on the internet.

8.2. Recommendations for Further Work

This thesis could be a foundation for other research and work on making solutions like OpenWISP more accessible. To list some of the possibilities of future work:

- Enhanced documentation: A dedicated effort to centralise and improve documentation, especially version-specific details, would significantly benefit new and less experienced developers. Throughout the development, this lack of centralisation has been a huge bottleneck.
- Stakeholder feedback Loop: Regular engagement with various stakeholders, can provide invaluable insights. Establishing a structured feedback loop can offer a continuous stream of suggestions for improvement.
- Security audits: With the mentioned dynamism of networks, regular security assessments are necessary and should not be overlooked.
- Engage with other open-source networking solutions: Looking into the possibility of integrating or collaborating with other open-source network management tools, learning from their successes and challenges. This could be very valuable, making smooth integrations between services.
- Developing the module marketplace: The Technical Design Document set the base for a feasible future development. However, it does not contain information on the specific data structures, for example. Future work on the design for this would be very valuable facilitating a future development of this webpage.

8.3. Closing statements and personal conclusion

With the development of the guide OpenWISP is more accessible for network administration users. Any technician looking for a solution to manage a network will be able to understand and take advantage of OpenWISP's potential for the most important infrastructure needs that organisations might have.

Moreover, the Technical Design Document on the module marketplace provides a general design on how to implement the module marketplace. However, this document would have been more relevant if done having knowledge of the specific available infrastructure for it, for example to define the Kubernetes Cluster with more details on the control plane and the nodes.

Finally, being able to work on a topic related to the courses that were studied the most at the university provided great satisfaction it was possible to use the resources acquired during the degree, putting them to work, having the opportunity to link this previous knowledge with comprehension of other perspectives, such as the needs of organisations. Nonetheless, the satisfaction is not only linked to the technical advancements but also to the conviction that the work done is meaningful to others. Furthermore, the fact that this was not a very collaborative project implied more independence, and although it was rewarding to take individual decisions without more debate than the internal one, it had its consequences. It implied less support, and this fact made a big difference to the point of view of team work for the researcher.

Bibliography

Ansible. (2019). How Ansible Works | Ansible.com. Ansible.com; Red Hat. https://www.ansible.com/overview/how-ansible-works

Barcelo, J. (2014, July 14). Bottom-up Broadband: Free Software Philosophy Applied to Networking Initiatives. https://github.com/jbarcelo/open_networks_paper/blob/master/bub.pdf?raw=true

Barcelo, J., Bellalta, B., Baig, R., Roca, R., Domingo, A., Sanabria, L., Cano, C., & Oliver, M. (2012). Bottom-up Broadband Initiatives in the Commons for Europe Project.

Battlemesh. (n.d.). Wireless Battle Mesh. Www.battlemesh.org. Retrieved June 9, 2023, from https://www.battlemesh.org/

Chetty, K., Qigui, L., Gcora, N., Josie, J., Wenwei, L., & Fang, C. (2018). Bridging the digital divide: measuring digital literacy. Economics: The Open-Access, Open-Assessment E-Journal, 12(23). https://doi.org/10.5018/economics-ejournal.ja.2018-23

Chris Sullo. (2021, May 5). nikto. GitHub. https://github.com/sullo/nikto

Cisco Meraki. (2020, October 5). Channel Planning Best Practices. Cisco Meraki. https://documentation.meraki.com/MR/Wi-Fi_Basics_and_Best_Practices/Channel_Planning_Best_Practices

Close The Gap . (n.d.). Close The Gap - Help us bridge the digital divide! Close-The-Gap.org. Retrieved February 23, 2023, from https://www.close-the-gap.org/

Cuevas, A. (2020, October 20). Qué es un Site Survey y para qué sirve. CUBE More than IT. https://integracion.cube.net.ar/blog/site-survey

David James Anderson. (2010). Kanban : successful evolutionary change for your technology business (pp. 13–16, 113–120, 139–144). Blue Hole Press.

Fortinet. (n.d.). What is Role-Based Access Control (RBAC)? Why is it Important? Fortinet. Retrieved May 27, 2023, from https://www.fortinet.com/de/resources/cyberglossary/role-based-access-control

Fortinet. (2023). What Is QoS (Quality of Service)? Fortinet. https://www.fortinet.com/resources/cyberglossary/qos-quality-of-service

Fundació Guifi·net. (n.d.). Red. Fundació Guifi·net. Retrieved June 14, 2023, from https://fundacio.guifi.net/es_ES/page/red

IBM. (2023, March 24). Virtual Local Area Networks . IBM. https://www.ibm.com/docs/en/aix/7.3?topic=cards-virtual-local-area-networks

Incommon. (n.d.). What is eduroam? InCommon. Retrieved March 16, 2023, from https://incommon.org/eduroam/what-is-eduroam/

Intel. (n.d.). 2.4 GHz vs. 5 GHz vs. 6 GHz: What's the Difference? Intel. Retrieved June 7, 2023, from https://www.intel.com/content/www/us/en/products/docs/wireless/2-4-vs-5ghz.html

LibreMesh. (n.d.-a). How it works. Libremesh.org. Retrieved May 25, 2023, from https://libremesh.org/howitworks.html

LibreMesh. (n.d.-b). LibreMesh. Libremesh.org. Retrieved May 23, 2023, from https://libremesh.org/

MikroTik. (n.d.). RouterOS license keys - RouterOS - MikroTik Documentation. Help.mikrotik.com. Retrieved April 23, 2023, from https://help.mikrotik.com/docs/display/ROS/RouterOS+license+keys

Nagios. (n.d.). Nagios Features. Nagios. Retrieved June 14, 2023, from https://www.nagios.org/about/features/

NSRC. (2017, January 27). eduroam and Identity Services. Www.youtube.com; Network Startup Resource Center. https://www.youtube.com/watch?v=PgWTBgmJetw

OECD. (2020). Enhancing Equal Access to Opportunities for Al. OECD Publishing. https://www.oecd.org/economy/Enhancing-equal-access-to-opportunities-OECD-background-note-for-G20-Framework-Working-Group-july-2020.pdf

OpenWISP. (n.d.-a). Architecture, Modules, Technologies — OpenWISP 22.05 documentation. Openwisp.io. Retrieved May 26, 2023, from https://openwisp.io/docs/general/architecture.html

OpenWISP. (n.d.-b). netjsonconfig — netjsonconfig 1.1a0 documentation. Netjsonconfig.openwisp.org. Retrieved June 12, 2023, from http://netjsonconfig.openwisp.org/en/latest/

OpenWISP. (n.d.-c). OpenWISP: OpenWRT Controller, public wifi, RADIUS, mesh networks. Openwisp.org. Retrieved May 25, 2023, from https://openwisp.org/whatis.html

OpenWISP. (n.d.-d). Values and Goals of OpenWISP — OpenWISP 22.05 documentation. Openwisp.io. Retrieved March 17, 2023, from https://openwisp.io/docs/general/values.html#goals

OpenWISP. (n.d.-e). WiFi Hotspot & Captive Portal — OpenWISP 22.05 documentation. Openwisp.io. Retrieved May 11, 2023, from https://openwisp.io/docs/tutorials/hotspot.html

openwisp. (2023). openwisp-monitoring. GitHub. https://github.com/openwisp/openwisp-monitoring

OpenWRT. (2023a). The UCI system. OpenWrt Wiki. https://openwrt.org/docs/guide-user/base-system/uci

OpenWRT. (2023b). Welcome to the OpenWrt Project. OpenWrt Wiki. https://openwrt.org/

Payscale. (2023a, February 18). Project Manager, Information Technology (IT). Payscale.com. https://www.payscale.com/research/ES/Job=Project_Manager%2C_Information_Technology_(IT)/Salary/7689aadc/Barcelona

Payscale. (2023b, February 18). Technical Writer Salary in Spain. Payscale.com. https://www.payscale.com/research/ES/Job=Technical_Writer/Salary/7d396eb3/Barcelona

Payscale. (2023c, February 27). Research Scientist Salary in Spain. Payscale.com. https://www.payscale.com/research/ES/Job=Research_Scientist/Salary/d192e53b/Barcelona

Quagga. (n.d.). Quagga (http://www.quagga.net). Quagga-Re.github.io. Retrieved May 11, 2023, from http://quagga-re.github.io/quagga-RE/

Red Eléctrica de España (REE). (n.d.). ESIOS Electricidad LUMIOS. Esios.ree.es; REE. Retrieved March 13, 2023, from https://www.esios.ree.es/es/lumios?rate=rate1&start_date=12-03-2023T18%3A06&end_date=13-03-2023T18%3A06&p1=0&p2=0&p3=1

Sumarno, S., Hartama, D., Gunawan, I., Tambunan, H. S., & Irawan, E. (2019). Optimization of Network Security Using Website Filtering With Microtic Routerboard. Journal of Physics, 1255(1), 012076. https://doi.org/10.1088/1742-6596/1255/1/012076

Team4Tech. (n.d.). Homepage. Team4Tech. Retrieved February 23, 2023, from https://team4tech.org/

Tien, F. F., & Fu, T.-T. (2008). The correlates of the digital divide and their impact on college student learning. Computers & Education, 50(1), 421–436. https://doi.org/10.1016/j.compedu.2006.07.005

TP-Link. (n.d.-a). What is Mesh WiFi? | Whole Home Mesh WiFi. TP-Link. Retrieved June 12, 2023, from https://www.tp-link.com/es/mesh-wifi/#b

TP-Link. (n.d.-b). Whole-Home Mesh WiFi. TP-Link. Retrieved June 12, 2023, from https://www.tp-link.com/en/mesh-wifi/#b

University of Wollongong. (2008, December 12). Advantages of VLANs. Documents.uow.edu.au. https://documents.uow.edu.au/~blane/netapp/ontap/nag/networking/concept/c_oc_netw_vlan-advantages.html

Wireshark. (n.d.). CaptureSetup/WLAN - The Wireshark Wiki. Wiki.wireshark.org. Retrieved June 11, 2023, from https://wiki.wireshark.org/CaptureSetup/WLAN

Zabbix. (n.d.). Zabbix features overview. Zabbix.com. Retrieved June 14, 2023, from https://www.zabbix.com/features?utm_campaign=mainpage&utm_source=website&utm_medium=header

Appendix A: OpenWISP Guide for Network Management in Community Environments

Prepared by Alicia Pallarol Isábal as part of a Bachelor thesis submission to Universitat Politècnica de Catalunya in October 2023.

Index:

A First steps **89**
A.1 Overview of OpenWISP Architecture 89
A.2 The starting point of the guide 91
B OpenWISP Customisation **93**
B.1 User Roles and Permissions 93
B.1.1 Network administration users 94
B.1.2 End-users 94
B.1.3 Access permissions 95
B.2 Network Usage Policies 96
B.2.1 Network Monitoring 97
B.3 Specific Module Configuration 98
B.3.1 Customising User Interfaces: Captive Page Interface 99
B.3.2 Workflow Automation: Onboarding Example 99
B.3.3 Inter-module Communication: Data Sharing and Task Triggering Example 100
B.4 Best Practices in Customisation 101
B.4.1 Adopting a Modular Approach 101
B.4.2 Prioritizing User Experience 101
B.4.3 Ensuring Network Security 102
B.4.4 Maintainability and Scalability 102
B.4.5 Testing and Validation 103
B.4.6 Compliance and Legalities 103
B.5 Automated Customisation Scripts 103
C Optimising Network Performance **107**
C.1 Understanding and Managing Network Traffic 107

C.1.1 Identifying High Traffic Periods 107
C.1.2 Identifying Bandwidth-Intensive Devices or Users 108
C.1.3 Managing Network Traffic 108
C.3.4 Troubleshooting Network Issues 108
C.2 Troubleshooting Common Network Issues 109
C.2.1 Installation/configuration issues 109
C.2.2 Device connectivity issues 110
C.2.3 Monitoring issues 110
C.2.4 Upgrade issues 111
C.3 Network Performance Monitoring 111
C.4 VPNs 111
D Ubiquitous Access and User-Centred Design 113
D.1 Designing for Ubiquitous Access 113
D.1.1 Network Layouts for Optimal Coverage 114
D.1.2 Device Compatibility and Access 115
D.2 User-Centred System Design 115
D.2.1 Creating User Guides for End-users 116
E Case Studies and Evaluations 117
E.1 The Value of Simulated Deployments 117
E.2 Performance Evaluations 118
E.3 Cost Analysis: OpenWISP vs Proprietary Solutions 118
E.4 Advantages of Open-Source Software 119
E.5 Methodological Framework for Continuous Evaluation 120
F Conclusion 121
F.1 Final thoughts 121
F.2 Next Steps and Further Learning 121
F.3 Additional Resources 122
F.4 Closing Remarks 123
G References 125

List of figures:

1. OpenWISP’s architecture diagram 89

2. OpenWISP dashboard with different modules shown in the side menu 92

Purpose of the Guide

As network infrastructure demands continue to evolve, especially within community settings, there is an increasing need to understand the tools that can best address these demands. OpenWISP, as a versatile Network Management System, emerges as a potential solution for diverse community scenarios, including the context of Hahatay, a Senegalese community.

This guide has two primary objectives:

1) To offer a thorough, yet approachable, exploration of OpenWISP's features and customisation opportunities. From adjusting user roles and configuring specific modules to understanding and applying best practices, this guide aims to present OpenWISP's functionalities in a digestible manner.
2) To showcase how OpenWISP can be applied in tangible real-world situations. Through in-depth evaluations, and strategies for network optimisation, readers can gain insight not just into the mechanics, but also the rationale behind each decision, highlighting OpenWISP's advantages over other solutions.

This guide is not trying to substitute OpenWISP’s documentation in any way, it tries to complement it. Actually, it contains links referring to either the documentation or the source code.

Who Should Read This Guide?

This guide is designed primarily for engineers, IT administrators, and other professionals interested in implementing and optimising OpenWISP for community network setups. While some sections delve into technical aspects, we have aimed to make the content accessible and valuable to those with a foundational understanding of networking concepts. If you are looking to leverage OpenWISP for your network management needs, and especially if you are considering its application in community environments, this guide is for you.

A First steps

A.1 Overview of OpenWISP Architecture

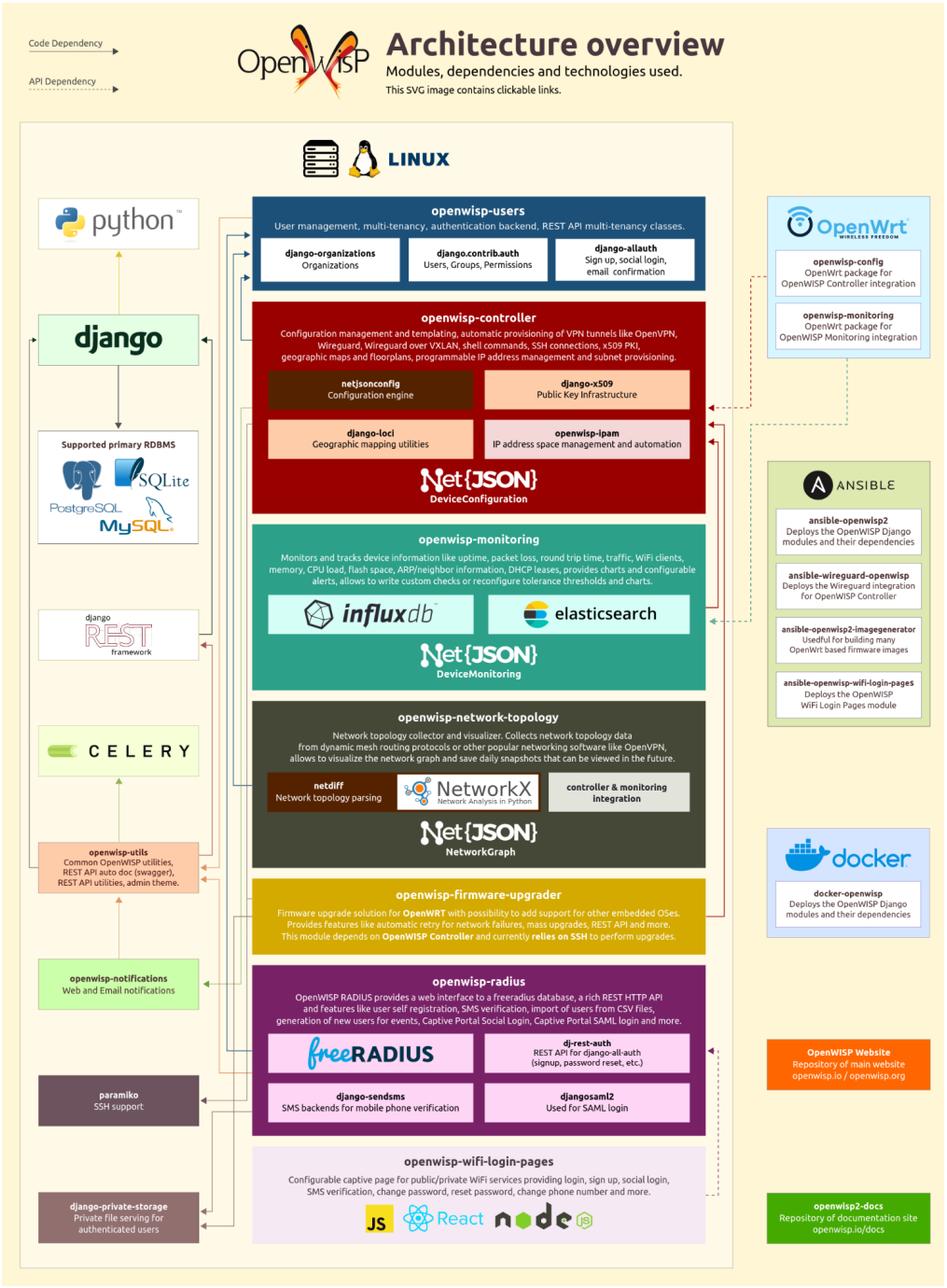

Figure 1[1]: OpenWISP's architecture diagram.

[1] Source: from OpenWISP (n.d-a).

The core of OpenWISP is the OpenWISP controller, which is a central server that manages and configures all the devices in the network (openwisp, 2023c). The controller is responsible for tasks such as provisioning new devices, configuring network settings, and updating firmware. It also provides a web-based interface that allows network administrators to view and manage the network. Moreover, OpenWISP is built on Django, therefore it benefits from the Django ecosystem. It can be extended using Django apps or even be integrated into larger Django projects.

Also, a database is required for an OpenWISP deployment, as shown in Figure 1. For this purpose, there are some RDBMS supported like PostgreSQL or MySQL. A centralized database stores all information. The different modules can have their specific tables or sets of tables within this centralized database. Each module interacts with the database to store and retrieve its specific data. For example, the monitoring module saves device performance metrics, while the controller saves device configurations and details.

But there are more modules other than the controller, such as the monitoring and network topology, for example. The OpenWISP controller interacts with the OpenWISP modules using a variety of different methods. One of them is Django signals, they allow certain senders to notify a set of receivers when some action has taken place. OpenWISP makes extensive use of signals. For instance, when a new device is registered in the OpenWISP controller, a signal might be emitted. The OpenWISP monitoring module, for example, can then catch this signal and initiate its own set of actions, like starting to monitor the new device and the network topology module can update the view of the topology accordingly.

However, there are other possibilities for module communication since the OpenWISP controller provides a REST API that can be used by the modules to communicate with each other. It also allows for integration with other systems, which means you can automate certain tasks or pull data from OpenWISP into other platforms.

As an example, one endpoint of the controller can be `GET /api/v1/controller/device/`, which lists the devices or this other endpoint `POST /api/v1/controller/device/{id}/command/` (openwisp, 2023c)

that executes a command in the device. Commands are specific actions that can be customised, there are two default ones.

As per communication, OpenWISP often uses established protocols and standards. For example, it uses the NetJSON format for network device configurations. This is an open standard, so it allows for interoperability with other systems that support NetJSON (OpenWISP, n.d.-a).

Furthermore, OpenWISP has a web-based interface, allowing administrators to manage and monitor devices from anywhere. This interface is built using modern web technologies, ensuring its responsiveness.

Because of its modular architecture, OpenWISP can be scaled as needed. If you only need certain features, you can only use those specific modules. You can add more modules as your network grows or your needs change.

A.2 The starting point of the guide

Once we are aware of OpenWISP's architecture, it is time to discuss at which point this guide supposes you have arrived when reading it. Also, the guide assumes the use of devices with OpenWRT firmware.

This guide is relevant once you have set up your OpenWISP server using Ansible, since we are aiming for a production environment. This set up is detailed here: https://github.com/openwisp/ansible-openwisp2#usage-tutorial.

The guide might require future updates due to the fact that soon enough a docker set up will also be available for production.

The guide discusses elements within the OpenWISP dashboard that may look similar to Figure 2:

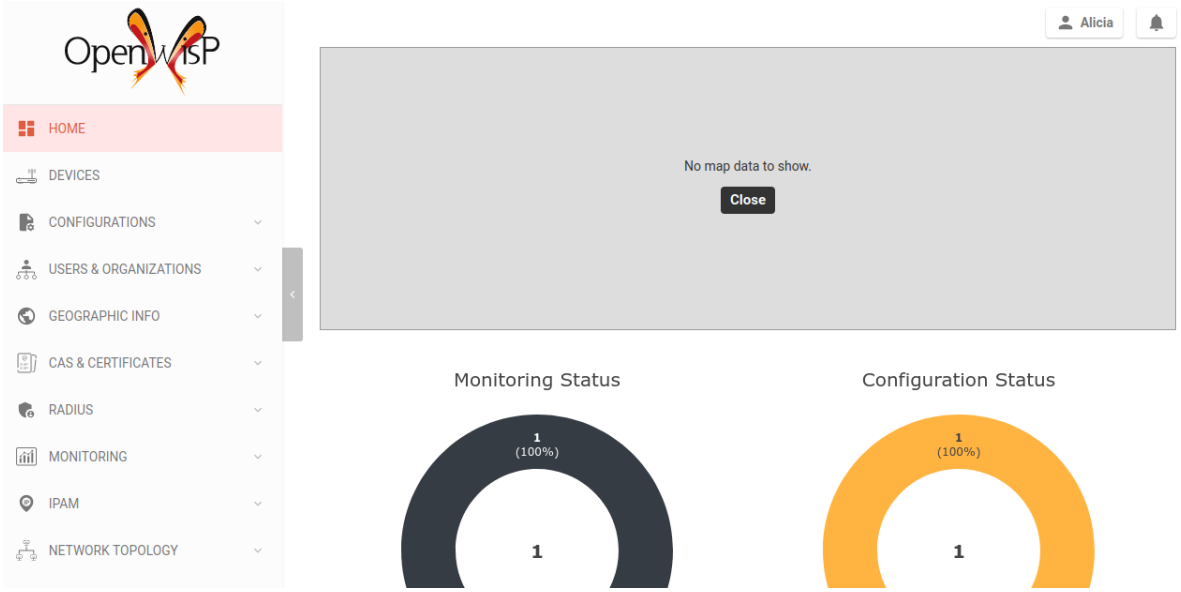

Figure 2[2]: OpenWISP dashboard with different modules shown in the side menu.

[2] Source: Screenshot of an own OpenWISP instance.

B OpenWISP Customisation

Customisation is a very broad concept when discussing OpenWISP because this platform has so many possibilities that there are many aspects to discuss. In OpenWISP you can customise every detail for every module, also the modules themselves are built to be independent of each other, so there are also a lot of customisation possibilities regarding the configuration of the whole system.

It is possible to change details such as the logo on the captive page for the RADIUS module. Furthermore, its open-source nature allows you to build another module extending an existing one or even create a completely different new one.

Since this topic is so broad and because building an entire module requires specific circumstances, we will not delve into every aspect of OpenWISP customisation. In this guide section, we will examine the most useful customisation options for general network infrastructures, discussing best practices. We will delve into some of the most important items to take into account in a community network and how to configure it using OpenWISP.

B.1 User Roles and Permissions

The platform categorizes users into two fundamental roles: network administration users, who oversee the setup, supervision, and care of the network, and end-users, the main beneficiaries of the network's connectivity. Each category has its distinct features and responsibilities.

For the network administration role, their responsibilities and access permissions are defined by their specific position within the network's hierarchy. Meanwhile, the end-user experience is related to RADIUS, a key networking protocol. Through RADIUS, the platform

can streamline the process of user verification, permission granting, and usage tracking, ensuring a secure and controlled network environment.

B.1.1 Network administration users

Some examples of network administration user roles we could define for our network infrastructure are:

- Administrator: This role has a high level of access, it has a broader oversight and organizational structuring. It can add users, groups, all kinds of configurations, etc. Essentially, manages the network from the organisation side rather than the technical one.
- Technician/operator: This role is for the staff that directly work with the network, it is for those who interact more frequently with the system. It has a great deal of access and its permissions are related to operational tasks.
- Auditor: This role is for specific staff that can evaluate the network. As for the access, they can view settings and logs, but not make changes.

In OpenWISP to create new roles and be able to link a user with the role, in the OpenWISP Dashboard we need to go to the lateral tab of *Users & Organizations* and then to the one named *Groups & Permissions.* There we should find that by default there is already an Administrator and an Operator group created. By clicking on *Add new group*, we can choose a name and pick from a list the permissions for this role.

B.1.2 End-users

As per the end-users, it is very interesting the level of customisation in the management that is achievable.

As in the network administration users, there are groups and the different users belong to a specific group. The groups are the customisable part, you can choose the belonging organisation, the group name and if it is the default assigned group to new users. But most importantly, you can set group replies and group checks these are ways to limit the bandwidth utilisation with attributes such as the maximum daily sessions and the maximum traffic per session. They come from the freeradius schema (OpenWISP, n.d.-b), freeradius is a deamon to set up a radius protocol (freeradius, n.d.).

By default, there are two groups “users” and “power-users”. The only difference is that users are the default group and they have limits.

B.1.3 Access permissions

In order for the users to be able to access the network, both network administration and end-users, need to be in the system.

For the end-users' case, they can be created in batches, from a .csv file but also one by one. OpenWISP can be configured so that users can also self-register using their mail or social accounts. Furthermore, paid plans can be implemented and be shown at the self-registration to have different limits to their connections. These payments are done via PayPal, more information can be found at https://openwisp.io/docs/tutorials/hotspot.html#paid-wifi-hotspot-subscription-plans.

The batch creation allows to only specify the mail of the user and the username, password and any related additional data is generated. It can be done in OpenWISP’s interface, in the Dashboard, in the tab Batch User Creation. There you can upload the .csv file, add a name and an expiration date for the new user. More information can be found at https://openwisp-radius.readthedocs.io/en/latest/user/importing_users.html.

The different groups of users can have different sets of permissions or limitations, customised as explained in the previous section. Some examples of groups of users that could be created are: students, teachers, staff and guests. Each one should have different limits.

The accounting, authorisation and authentication are done with the package Coova-Chilli, which provides the captive portal interface (OpenWISP, n.d.-d).

B.2 Network Usage Policies

Since a community network has many different spaces with different purposes and therefore needs, it is important to formally define the usage policies for the network with the specification of network administration roles and permissions, end-user roles as well as the justification behind the collected data.

This is important because it smoothes the standardization of processes; facilitates the changes in the technical staff; and can help future policymaking because of the existence of context. Moreover, these policies facilitate security revisions, for example, user privileges.
OpenWISP is not just a tool to manage networks but a means to implement, enforce, and monitor adherence to network usage policies in a systematic and automated manner. In this section, we will discuss the most relevant policy and network-wise topics we can execute with OpenWISP.

- Access: in the previous section, the possibilities to implement different access types for new users were discussed. To have a secure network, we need to decide how will the users access the network. For how to do it, we can find it at *section B.1.3*.

- Resource allocation: in user management, policies indicating the limits for each group can help guarantee QoS as far as bandwidth goes, as explained in *section B.1.2*.
- Monitoring: monitoring a network, gathering data on the users and the usage, can help identify inadequate usage, all in all, ways to improve the policies and to make sure that they are respected. Also, in terms of policy definition, it is important to define the consequences of policy violation. Since monitoring in OpenWISP is a broad topic, we will discuss it in the next section.

B.2.1 Network Monitoring

OpenWISP can track each user and device. For the devices, at the dashboard there is a Monitoring tab, there we can find two more tabs: Metrics and Checks.

The default metrics can be found at: https://github.com/openwisp/openwisp-monitoring/blob/16a6608222658295ba6d0bd4dcd848a36e94b308/openwisp_monitoring/monitoring/configuration.py#L49C20-L49C20. They are:

- Ping: It checks if a specific device is reachable using fping.
- Configuration applied: It checks if a specific configuration has been applied to a specific device.
- Traffic: It measures the network traffic of a specific interface in GB. Both download and upload.
- Clients: It lists the Wi-Fi clients associated with a specific interface.
- Disk: It checks the disk usage of an available partition of a specific device.
- Memory: It checks the percentage of RAM being used by a specific device.
- CPU: It checks the average CPU load of a specific device while accounting for the available CPUs. It is measured using the Linux load averages.

- Signal strength: It measures the signal strength and signal power in dBm.
- Signal quality: It measures the Signal to Noise Ratio in dB.
- Access technology: It lists the access technology being used by the device.

It is possible to add new metrics and to change the tags or any fields within them. Most of these metrics send emails or notifications of some kind whenever a certain value is reached, for example. Also, it is possible to define additional charts or to override the default charts for the metrics, for instance, if a metric was measured in GB it is possible to change the measurement in the chart showing it in MB. There are examples for this following this link: https://github.com/openwisp/openwisp-monitoring/tree/1.0#openwisp_monitoring_metrics.

As per the available checks, there are three: ping, configuration applied and Iperf3. The first check looks for information on both the uptime and the RTT (Round trip time), the uptime, packet loss and rtt charts are created. The configuration applied is a periodic check that ensures changes to configurations are performed in time, it is also triggered whenever the configuration status of a device changes, so the flow of information is seamless. Finally, the Iperf3 check looks for information directly related to the performance, for example the jitter and the datagram loss (openwisp, 2023d).

B.3 Specific Module Configuration

There are some modules that need a lot of customisation or some tasks within those modules that require specific customisation, in this section, we will discuss the most relevant ones.

B.3.1 Customising User Interfaces: Captive Page Interface

The captive page interface is vital because it is often one of the first points of interaction between the end-user and the network. OpenWISP provides options to tailor this to different user bases in many ways:

- Multilingual Support: OpenWISP can ensure that the captive page is available in different languages based on the location of the infrastructure so that the end users can choose. In order to change this in OpenWISP, we need to check the documentation at https://github.com/openwisp/openwisp-wifi-login-pages#defining-available-languages where it is explained.
- Branding: Integrating organization or network branding into the captive page, which might involve logos, colour schemes, and typography that adhere to a particular brand identity. In order to change the logo you can follow the instructions here: https://github.com/openwisp/openwisp-wifi-login-pages/tree/1.0#variants-of-the-same-configuration. Note the importance of not deleting the default .yaml file.
- Access Control: as discussed in *section B.1.3*, there are different access flows possible for a user.

B.3.2 Workflow Automation: Onboarding Example

Automating workflows is necessary in order to be efficient, for this reason, we must consider the task of onboarding new devices or users onto the network. An automated workflow could on the one hand manage the onboarding of new devices and onboarding of new users.

For the devices, with the NetJSON configuration templates and automatic registration, it is possible to fully automate the deployment of new nodes in your network (openwisp, 2023a)

(openwisp, 2023b). Once the templates are configured as needed, it is only necessary to register a device and it will autoconfigure according to the template.

As per the users, if we accepted self-registration, the creation of new users would be streamlined as well as the assignment and management of user credentials, potentially integrating with existing user management systems or databases, using the Users module, calling its API.

In *section B.5* we find an example of a script that can serve as a base for automating configuration for devices.

B.3.3 Inter-module Communication: Data Sharing and Task Triggering Example

Inter-module communication is about how one module within OpenWISP can use or trigger functionality in another. As we already discussed, there are certain metrics that trigger notifications to the network administrators. However, there are other events where task triggering is interesting.

For example, the OpenWISP firmware upgrader module can interact with OpenWISP notifications module. When a firmware upgrade is performed using the Firmware Upgrader module, it can trigger a notification in the Notifications module. This notification could inform the network administrators about the status of the firmware upgrade, such as whether the upgrade was successful or if there were any issues. This would be done by extending the firmware upgrader module and adding the necessary API calls. Since this is a very specific topic more focused on general web development, we will not dig deeper into it.

B.4 Best Practices in Customisation

An infrastructure is always evolving, for this reason, it is important to follow best practices guidelines in order to be perfectly aligned with the network's needs. For this reason, in this section, we will be discussing some of these best practices in relation to an infrastructure that uses OpenWISP.

B.4.1 Adopting a Modular Approach

In order to make the most out of OpenWISP's modular architecture, it is required to strategically choose and implement the modules that are actually needed in our infrastructure.

The fact is that we should only implement modules that are directly beneficial to avoid unnecessary complexity. For instance, the OpenWISP firmware upgrader module might not be necessary for smaller infrastructures manually managed, so it would be best to skip it. But at the same time, we should be open to exploring how additional modules can be integrated as the network grows or demands evolve.

B.4.2 Prioritizing User Experience

Given that networks have many users, we should always keep in mind who is using the network in order to make all features accessible to the variety of user needs and abilities. This topic is highly related to the intuitiveness of the interface of the captive page, for example, where it is important to support different languages and to customise it by adding a logo, so the users can easily associate the captive page with the network.

B.4.3 Ensuring Network Security

OpenWISP allows for secure provisioning and configuration management of devices. It supports any type of configuration supported by OpenWRT. Moreover, it includes a x509 PKI, management system, using the django-x509 django app (openwisp, 2023c). This allows for the secure authentication of devices on the network.

The secure communication protocols that OpenWISP supports secure communication protocols. For example, it uses SSH access for push updates.

Furthermore, an important item in terms of security is having a Role-Based Access Control (RBAC) policy, where no user has more access than he or she needs. This is essential to limit human errors. For this reason, defining each network administration user role with the detailed access level it needs is essential.

Finally, the RADIUS module provides support for RADIUS, a networking protocol that offers centralized Authentication, Authorization, and Accounting (AAA) management for users who connect and use a network service. This includes support for WPA Enterprise (802.1x), captive portal authentication, and more (openwisp, 2023e).

B.4.4 Maintainability and Scalability

As we develop our infrastructure and tailor OpenWISP to our needs, the customisation should be undertaken with future maintainability and scalability in mind.

To maintain it, we need to be thorough with the documentation, registering all about our infrastructure's customisations, configurations and practices for future reference and troubleshooting. Not only that, but we also need to make the most out of the resources and the capabilities of OpenWISP, using templates for new devices to ensure consistent

configuration among different devices, saving time for the administrators. Moreover, its modular architecture allows the system to scale with the network's needs.

B.4.5 Testing and Validation

The open-source nature of OpenWISP that allows us to extend modules based on our needs can be dangerous because, with our own development, we could be introducing security threats or performance downgrades for the network.

In order to prevent this, we must always have pre-deployment testing, where we validate custom features in a controlled environment before deploying them into the live network. We must also have continuous monitoring to be aware of the impacts of customisations on network performance and security, conveniently for this we can use the built-in checks and alerts in the monitoring module to validate and monitor custom configurations.

B.4.6 Compliance and Legalities

In each country, compliance can be very different. This is why we must ensure that customisations and network management practices comply with relevant laws, regulations, and policies.

OpenWISP can gather various data from the user and we must know which one we can use or store and which one is sensitive. On the user privacy topic, one of OpenWISP's core goals is that user personal data is never sold to third parties and that traffic logs are stored only for the period of time mandated by law (OpenWISP, n.d.-c).

B.5 Automated Customisation Scripts

Here is an example of a basic configuration script to create a new configuration in OpenWISP. In this context, a configuration is a set of parameters that can be applied to a

device to configure its network settings. Configurations can be used to configure a wide range of settings, such as the device's IP address, subnet mask, gateway address, DNS servers, and wireless network settings.

As mentioned this is a basic Python script so it does not include error handling, it was elaborated specifically for this guide by Pallarol (2023):

```
import requests
import json

OPENWISP_SERVER = 'https://your-openwisp-server.com'
USERNAME = 'username'
PASSWORD = 'password'

CONFIG_NAME = 'New Network Configuration'
CONFIG_BACKEND = 'netjsonconfig.OpenWrt'
DEVICE_NAME = 'New Device'
DEVICE_MAC_ADDRESS = '00:00:00:00:00:00'

LOGIN_ENDPOINT = f'{OPENWISP_SERVER}/admin/login/'
CONFIGURATION_ENDPOINT = f'{OPENWISP_SERVER}/api/v1/config/'

login_response = requests.post(
     LOGIN_ENDPOINT,
     data={'username': USERNAME, 'password': PASSWORD},
     headers={'Content-Type':
'application/x-www-form-urlencoded'},
     allow_redirects=False
)

if login_response.status_code == 302:
     csrftoken = login_response.cookies['csrftoken']
```

```
sessionid = login_response.cookies['sessionid']

headers = {
'Content-Type': 'application/json',
'X-CSRFToken': csrftoken,
}

cookies = {'sessionid': sessionid}

config_payload = {
"name": CONFIG_NAME,
"backend": CONFIG_BACKEND,
"config": {
     # specific configuration parameters
},
"device": {
     "name": DEVICE_NAME,
     "mac_address": DEVICE_MAC_ADDRESS,
}
}

config_response = requests.post(
CONFIGURATION_ENDPOINT,
data=json.dumps(config_payload),
headers=headers,
cookies=cookies
)
if config_response.status_code == 201:
print("Configuration created successfully!")
else:
```

```
    print("Failed to create configuration. Reason:", config_response.text)
else:
    print("Login failed. Please check your credentials.")
```

This script logs in to the OpenWISP server and obtains a cookie for subsequent requests. Then it prepares a configuration payload with the specific configuration parameters, such as the name, backend, config parameters, device name, and MAC address. After this, it posts the configuration payload to the OpenWISP API endpoint to create the new configuration and finally it prints a success message if the configuration was created successfully, or an error message if it failed.

Please replace 'https://your-openwisp-server.com', 'username', 'password', 'New Network Configuration', 'New Device', and '00:00:00:00:00:00' with the actual server details, username, password, configuration name, device name, and device MAC address respectively.

Even though you could simply use the OpenWISP Dashboard to apply a configuration, a script like this one can help automate the configuration for new devices and would be specially useful for registering many new devices at once or for complex configurations.

C Optimising Network Performance

To have a well-functioning network optimisation mechanisms are crucial. They enable the optimal network operation. In this section, we will discuss some of the most important network performance topics to keep in mind while using OpenWISP.

Some topics we will not be discussing are channel selection and VLAN management. Even though these topics are undoubtedly relevant to a network infrastructure OpenWISP does not have a functionality related to them, we would need to use 3rd party applications and that is not the object of this guide.

C.1 Understanding and Managing Network Traffic

This section is based on the premise that we are familiar with the monitoring module at https://github.com/openwisp/openwisp-monitoring and all its default metrics since we will be learning about the different ways we can use these metrics and discussing how to make the most out of our network. We will be breaking down the importance of specific metrics so that if we consider it necessary we can add notifications or alerts as explained in *section B.2.1*.

C.1.1 Identifying High Traffic Periods

With the data from the traffic metric, administrators can scrutinize temporal patterns in the network. By identifying periods of high traffic, institutions can strategically plan network-intensive tasks like updates or backups during off-peak hours, ensuring that the network performance remains unhampered during crucial learning hours, for example.

C.1.2 Identifying Bandwidth-Intensive Devices or Users

OpenWISP provides insights into the utilization of network resources per device with the metrics. This becomes pivotal in identifying bandwidth-Intensive devices or users. By analysing the metrics, data visualization tools and logs, administrators can pinpoint devices that recurrently showcase excessive bandwidth usage. This can also aid in recognising unauthorised usage or malware-infected devices which might be silently consuming bandwidth, providing a layer of network security through vigilant monitoring.

C.1.3 Managing Network Traffic

Utilising the metrics from OpenWISP's monitoring module allows network administrators to implement data-driven strategies for managing network traffic. In the context of Quality of Service (QoS) rules, data like packet loss, jitter, and latency can be observed and analysed through OpenWISP, which can subsequently inform the configuration of QoS rules to prioritize certain types of traffic, ensuring that critical educational platforms are always accessible and performant for users.

Furthermore, with OpenWISP, administrators can set alerts to be triggered when certain bandwidth thresholds are met or exceeded, enabling proactive management before issues escalate. In a practical scenario, based on usage data, administrators might decide to impose bandwidth limits or block non-educational traffic during learning hours to safeguard the network’s performance for educational activities.

C.3.4 Troubleshooting Network Issues

The monitoring module can serve as a valuable asset in identifying and diagnosing network issues. For instance, administrators might observe inconsistencies or anomalies in the traffic data, which could be indicative of various issues. A sudden and unexpected drop in network traffic might signal a network outage or failure of a critical network component. On

the other hand, a gradual reduction in speed over time might suggest hardware degradation or issues with the internet service provider.

OpenWISP enables administrators to not only visualize this data but also to receive alerts for specific incidents. The historical data provided by the monitoring module ensures that administrators can backtrack and identify when an issue may have originated, significantly reducing the troubleshooting time and ensuring rapid resolution.

C.2 Troubleshooting Common Network Issues

According to the experimentation carried out in the thesis that contains this guide, there are some issues when configuring OpenWISP. They will be discussed in this section.

The end goal of this section is to encourage you to ask in the OpenWISP channels. There you will be able to find help or people willing to help you: https://openwisp.org/support.html.

C.2.1 Installation/configuration issues

This guide, as previously mentioned, is supposed to be used once we have a full server deployment of OpenWISP. However, there might be some issues with dependencies, environment setup, or database configuration.

For the dependencies, you should ensure that all the required ones are installed. For the environment set-up, make sure that the system meets the minimum requirements for running OpenWISP; if you are using a virtual environment, check that it is correctly set up and activated. Moreover, check both the environmental variables required and the permissions of the user account that you are using.

C.2.2 Device connectivity issues

This issue could be due to network problems, incorrect device configuration, or compatibility issues.

- If the device is not connecting to the OpenWISP server: you should check the device's network connection, check the configuration of the device related to the server connection and also verify that the device's firewall or security settings are not blocking the connection.
- If the device does not appear in OpenWISP's interface: you should in a first instance wait, since it might take some time for a newly connected device to appear. If the device still does not appear after a while, check the device's configuration and ensure it is correctly set up to connect to the OpenWISP server.
- If the device is showing as disconnected in OpenWISP's interface: check if the device is turned off, disconnected from the network, or experiencing network issues.
- If the device is not sending monitoring data: ensure the monitoring module is correctly installed and configured. Also, check the device's network connection and if you are still not receiving monitoring data, there might be an issue with the monitoring configuration on either the device or the OpenWISP server.

C.2.3 Monitoring issues

Setting up or interpreting the monitoring module might cause problems. This could include issues with collecting data from devices, displaying the data in the OpenWISP interface, or setting up alerts.

If there were issues with the data collection, it could be a configuration issue of both the device and the server or just an issue with the connectivity of the device. However, if data was not displaying correctly it can be due to the fact that they need some time. Finally, if

there is an issue with the alerts, firstly, check the settings but afterwards, again, there is a configuration issue.

C.2.4 Upgrade issues

You might also encounter problems when upgrading OpenWISP or its modules. This could be due to incompatible versions, database migration issues, or changes in the system requirements.

This issue is based on prevention rather than solutions because these errors could be avoided by good documentation, especially. But also, with attention to the versions, keeping track of them.

C.3 Network Performance Monitoring

To monitor network performance OpenWISP provides the already discussed monitoring module, where all the checks and metrics are available. As it was discussed, it is possible to set notifications depending on specific performance metrics.

To make the most out of this module and if we do not have previous data on the performance, here are two strategies: to either define the checks and notifications predicting the network usage data, or to gather data about usage and then define the checks and notifications.

C.4 VPNs

You can host a VPN server on the same machine where the rest of the OpenWISP services are running. For this reason, it is important to discuss the optimisation when hosting a VPN server and when using VPNs in general. In this section, we will discuss general items to keep into account when setting VPNs to ensure the best operation.

- System resources: Running a VPN server can consume significant system resources, especially if you have many devices connecting to it. For this reason, you should make sure your server has enough resources (CPU, RAM, network bandwidth) to handle the additional load. All those can be checked with OpenWISP's monitoring feature. Once the server is set up, it is also important to check the resources' management, so with the monitoring module as well, we can keep track of the performance with metrics like connection speeds, packet loss rates or server load.
- Automating VPN configuration: OpenWISP can automate the process of configuring VPN tunnels, saving time and reducing the risk of errors compared to manually configuring each device. For automating the VPN configuration, the steps are:
    - Once you have the controller module installed, you can create a VPN server in the OpenWISP dashboard. During this process, you will need to import the CA (Certificate Authority) and the Server Certificate.
    - Then, still in the dashboard, you can create a configuration template for VPN clients. This template will be used to automatically generate the VPN configuration for each device.
    - Finally, if you enable by default a configuration template, it will be automatically assigned to new devices. This means that when you register a new device in OpenWISP, it will autoconfigure itself based on these default templates.
- Network configuration: Depending on your network configuration, you might need to adjust firewall rules or port forwarding settings to allow VPN connections to your server.

Also, running a VPN server on the same machine as your other services can have security implications. Make sure to follow best practices for securing your server such as keeping software up-to-date, using strong encryption settings, and regularly reviewing access logs.

D Ubiquitous Access and User-Centred Design

In this section we will discuss how OpenWISP is centred around the access and the user. It is aimed at low-cost networks, including public Wi-Fi, university Wi-Fi, mesh networks, and IoT. But some features that describe it more accurately are:

- Support for various devices and network layouts: OpenWISP allows you to set up any type of configuration supported by OpenWRT, thanks to its advanced mode. This means it can support a wide range of devices and network layouts. Also, thanks to the network topology module, it is possible to visualise the network operating.
- User-friendly interface: OpenWISP provides a network management interface that may present a learning curve for administrators. The platform is indeed robust and feature-rich, but acknowledging that experiences with the interface can be diverse, this guide is here to assist in navigating through that learning curve effectively.
- Multi-tenancy support: OpenWISP Users module provides user management, multi-tenancy, authentication backend, REST API utilities, and classes to implement multi-tenancy.

Other topics that showcase this are the monitoring module, the automation possibilities and the modularity, all these items together, make network management easier and that is all due to the fact that OpenWISP is designed to be focused on network access but with the user in mind.

D.1 Designing for Ubiquitous Access

The items we just mentioned prove OpenWISP to be designed for ubiquitous access, however, in terms of access, it is worth emphasising the network topology and layout, and

the variety of devices that can work with OpenWISP. For this reason, we are going to discuss these topics further.

D.1.1 Network Layouts for Optimal Coverage

OpenWISP's design allows it to support various network layouts for optimal coverage by collecting and processing network topology data from different networking software, providing flexible data collection strategies, offering real-time link status monitoring, and visualizing the network topology. Here are some further specifics on how does OpenWISP do all that:

- Network topology collector and visualizer: the OpenWISP Network Topology is a complete web application and API that allows to collect network topology data from different networking software, including dynamic mesh routing protocols and OpenVPN; which means it can support various network layouts. It stores this data, visualizes it and allows users to edit its details.
- Topology data collection strategies: OpenWISP supports two strategies for collecting topology data: FETCH and RECEIVE. The FETCH strategy involves fetching topology data from a specified URL, while the RECEIVE strategy involves receiving topology data via POST API requests.
- Topology data processing: OpenWISP uses parsers to process topology data based on the selected topology format. This allows it to handle different types of network topologies.
- Link status change hooks: OpenWISP provides hooks to execute code when the status of a link changes. This can be useful for monitoring the network and responding to changes in real time.

More details can be found in the Github repository for the network topology module at: https://github.com/openwisp/openwisp-network-topology.

D.1.2 Device Compatibility and Access

OpenWISP is designed to be compatible with a wide range of devices, particularly those running OpenWRT, actually it is compatible with any device as long as the device supports OpenWRT. This includes a wide range of routers and other networking devices. This is because OpenWISP uses the openwisp-config package, which is the controller agent for OpenWRT that is within the controller module.

The process of installing openwisp-config on an OpenWRT device is straightforward. It involves flashing the device with OpenWRT, enabling SSH access, connecting the device to the internet, and then installing openwisp-config, as you may have already done at this point in the guide either with physical devices or with virtual machines.

As for secure access control, OpenWISP provides this through its user management module which we already mentioned. This module provides features such as management of users, organizations, and permissions; multi-tenancy support so multiple organizations or departments can use the same instance of OpenWISP while keeping their data separate and the module also provides an authentication backend that can be used to authenticate users. Furthermore, the module provides REST API utilities that can be used to interact with the user management system programmatically.

D.2 User-Centred System Design

The interface of OpenWISP, while functional, can be complex and overwhelming for new users. The system's reliance on OpenWRT configurations and its modular nature can make it challenging to navigate and use effectively. This complexity can create a steep learning curve.

It is also an important discussion of the need for end-user guides and the approach to building one depending on the needs of the infrastructure, so the usage is enhanced in aspects ranging from practical use to the network's security.

Despite these challenges, OpenWISP's flexibility and extensive capabilities make it a powerful tool for network management. With the right guidance and resources, users can leverage these features to implement comprehensive user guides as we will be discussing in this section.

D.2.1 Creating User Guides for End-users

End-users need clear, accessible information on how to use the network services effectively, comply with usage policies, and troubleshoot common issues. For this, creating user guides tailored to end-users is indeed crucial. These guides should focus on:

- Usage Instructions: Clear, step-by-step instructions on how to connect to the network, use the services provided, and make the most out of them. The instructions should clarify the registering options, if they can self-register or if they will be sent a password and username, for example.
- Policy Compliance: An easy-to-understand summary of the network's usage policies, including any data limits, restricted activities, and fair usage policies.
- Troubleshooting Guide: Simple solutions to common problems that users might encounter while using the network services, such as what to do or who to contact if they can not connect to the network or if their connection is slow.

In conclusion, while OpenWISP may not provide out-of-the-box user guides or centralised documentation suitable for all users, it is possible for organisations to create effective user guides that help understand how to use the network's features effectively and efficiently.

E Case Studies and Evaluations

Navigating through the deployment and management of OpenWISP necessitates a thorough understanding that spans beyond technical proficiency. Therefore, this section seeks to explain key aspects such as the design of case studies, performance evaluations, cost analysis, and continuous evaluation methodologies.

Integrating these components into a guide will not only help open OpenWISP to a broader audience but also facilitate a robust understanding of how to implement, manage, and optimise an OpenWISP-based network effectively within real-world conditions.

E.1 The Value of Simulated Deployments

Simulated deployments are an invaluable tool in network management, particularly when setting up and optimising network configurations. They allow network administrators to test and fine-tune their configurations in a controlled environment before deploying them in a live environment. This can help identify and address potential issues, ensuring a smoother and more reliable network operation. OpenWISP's configuration management features support this kind of testing because it has OpenWRT compatibility, so Virtual Machines can be used. Also, the NetJSON configuration templates that OpenWISP uses allow you to define reusable network configurations that you can apply in your simulated environment to test their performance and impact on the network.

Moreover, the automatic registration and in general the automatisation allow you to simulate the environment accurately enough to easily scale your testing to include more devices without manual configuration.

Finally, OpenWISP’s modularity allows you to add or modify features as needed to suit your testing requirements.

E.2 Performance Evaluations

Evaluating the performance of networks managed by OpenWISP involves several aspects, including monitoring network traffic, identifying high-usage devices, and troubleshooting connectivity issues. Here's how OpenWISP supports these tasks:

- Monitoring network traffic: OpenWISP Monitoring is a network monitoring system is the module that collects the important metrics for performance evaluations like device status information like uptime, RAM status, CPU load averages, interface properties and addresses, among others.
- Identifying high-usage devices: as we mentioned in *section C.1.2*, OpenWISP can help identify high-usage devices.
- Troubleshooting connectivity Issues: the monitoring features can also help troubleshoot connectivity issues. For example, monitoring uptime and packet loss can help identify devices that are frequently disconnecting or experiencing high packet loss.

Collecting and analysing network data, OpenWISP allows network administrators to monitor network traffic, identify high-usage devices, and troubleshoot connectivity issues.

E.3 Cost Analysis: OpenWISP vs Proprietary Solutions

OpenWISP, as an open-source network management system, offers several cost benefits compared to proprietary solutions. Here are some key factors:

- No licensing costs: Unlike proprietary software, open-source software like OpenWISP is free to use. This means organizations can deploy and use OpenWISP without worrying about licensing costs.

- Customisation capabilities: OpenWISP is highly customisable as we discussed in *chapter 1*. It allows organizations to modify and extend its functionality to suit their specific needs. In contrast, proprietary solutions often come with rigid features and limited customisation options.
- Community support: OpenWISP has a strong community of developers and users. This community can provide valuable support and resources, helping organizations solve problems and improve their network management. Proprietary solutions, on the other hand, usually offer support at an additional cost.
- Avoid Vendor Lock-in: With open-source software like OpenWISP, organizations avoid vendor lock-in that often comes with proprietary software. They have the freedom to modify the software or switch to a different solution if needed.

E.4 Advantages of Open-Source Software

Apart from the no licensing costs, avoiding the vendor lock-in, the community support and the ability to be extensible, there are more advantages of using open-source software:

- Transparency: Open-source software is transparent. The source code is openly available, which allows users to inspect it, understand how it works, and make modifications if needed.
- Attract better talent: Companies that use open-source software can attract better talent since some developers may prefer organizations that contribute and use open-source software.
- Better security: The transparency of open-source software allows anyone to inspect the code, which can lead to more secure software. Security gaps are quickly tracked down and closed under the inspection of many eyes.

E.5 Methodological Framework for Continuous Evaluation

With the performance evaluations we do not have enough. We need to have continuous evaluation. So apart from the performance ones, we should also take into account that there must be periodic audits of network security to ensure that the network is secure and that all security measures are working as expected.

Also, continuous feedback from network users can provide valuable insights into how the network is being used and where improvements can be made. While OpenWISP does not provide a built-in feature for collecting user feedback, organizations can use various methods such as surveys, user interviews, or feedback forms to gather this information.

Finally, it should be interesting to use continuous improvement methodologies. An example is the Kaizen methodology (United States Environmental Protection Agency, 2023), it focuses on continuous improvement through incremental changes. In the context of network management, this could involve regularly making small adjustments to the network configuration or policies based on performance data and user feedback.

F Conclusion

F.1 Final thoughts

This guide unfolds the significant capabilities of OpenWISP, outlining its potential in managing network infrastructures. It brings forth invaluable functionalities, notably RADIUS, which facilitates centralized AAA (Authentication, Authorization, and Accounting) management. OpenWISP, with its open-source nature, is not merely a tool; it is a versatile platform, a perfect fit to a spectrum of infrastructures thanks to its modularity and expansive customisation possibilities.

The dynamism embedded within OpenWISP not only empowers network administrators to optimise, secure, and adeptly manage their networks but also necessitates an appreciation for its evolutionary nature. Particularly, during the crafting of this guide, OpenWISP introduced production capabilities via Docker, spotlighting the importance of being proactive, informed users. Staying tuned to such advancements ensures that we harness OpenWISP's full potential and remain adept in managing evolving network demands.

F.2 Next Steps and Further Learning

Embarking on your implementation journey with OpenWISP, you will unearth a variety of strategies and customisations, each paving the way towards seamless network management possibilities. The unique demands of every network request the adaptation of these strategies, culminating in the gradual integration of advanced functionalities as your expertise blossoms.

Venturing into advanced customisations, including scripting and API integration, reveals new horizons in tailoring your network management solutions. While you are crafting a

management ecosystem that aligns with your network's dynamics, remember that thorough testing and validation are crucial to maintain network security and integrity.

Advanced customisations might entail developing tailored modules to cater to specific network management needs, automating network configuration adjustments in response to real-time monitoring data, or integrating OpenWISP with third-party applications and platforms via APIs to enhance data sharing and functionality.

Immersing yourself in OpenWISP not only represents a step towards adaptable, scalable, and proficient network management but also a contribution to its collective evolution, ensuring its position as a future-ready solution. Your innovations and experiences do not just strengthen your own network infrastructure, they enrich the entire OpenWISP community and platform.

Engagement with this community through forums and discussion channels offers an avenue for perpetual learning, problem-solving, and the exploration of innovative applications, turning challenges and transformations into opportunities for growth and exploration.

F.3 Additional Resources

Navigating through the complexities and specificities of network management with OpenWISP can be significantly enriched by diving into a myriad of additional resources available. The official documentation and various tutorials from OpenWISP furnish a foundational base, providing an in-depth exploration of functionalities, modules, and customisation possibilities. These materials, accessible through the OpenWISP website, serve not only as a guidepost for troubleshooting but also as a treasure trove of knowledge to master the application of this versatile platform.

An exploration into real-world applications of OpenWISP is facilitated by various case studies, each narrating a unique tale of network management challenges and innovative

solutions. These instances not only unveil the practical applicability of strategies discussed in this guide but also illuminate the diverse ways in which OpenWISP adapts to distinct network environments, becoming an invaluable asset in navigating through myriad network management scenarios.

For those who wish to delve deeper into specific topics or explore new terrains in network management, the extensive OpenWISP documentation is available and we invite you to explore it. These writings not only expand upon the concepts and strategies discussed within this guide but also explore new ideas, advanced customisations, and innovative applications within the expansive universe of OpenWISP. Thus, your journey through network management with OpenWISP is accompanied, at every step, by knowledge and experiences shared by a global community.

F.4 Closing Remarks

Thank you sincerely for embarking on this journey through network management with OpenWISP, and investing your time and energy into exploring this guide. Your endeavour into enhancing and innovating your network's functionality and efficiency is very significant not only for your immediate community but it also adds value to the global network of OpenWISP users and administrators. We encourage you to dive into the practical applications of strategies and customisations discussed herein, as each network presents a unique landscape, ripe for exploration and tailored applications of OpenWISP's diverse capabilities.

Your experiments, insights, and experiences with implementing OpenWISP are invaluable, and as you traverse through the myriad possibilities, do remember that your journey enriches not only your network but can also inspire pathways for others in the community. Your feedback on this guide and contributions to the OpenWISP community, whether through

sharing your own narratives, crafting more guides, or developing modules, are not only welcomed but deeply appreciated.

In navigating through the challenges and discoveries that await in your network management journey, may you find innovative solutions and explore new territories in network customisation and management with OpenWISP.

G References

freeradius. (n.d.). guide/FAQ. Wiki.freeradius.org. Retrieved September 3, 2023, from https://wiki.freeradius.org/guide/FAQ#freeradius-overview_what-is-freeradius-and-what-is-it-supposed-to-do

OpenWISP. (n.d.-a). Architecture, Modules, Technologies — OpenWISP 22.05 documentation. Openwisp.io. Retrieved May 26, 2023, from https://openwisp.io/docs/general/architecture.html

OpenWISP. (n.d.-b). Enforcing session limits — openwisp-radius 1.0.2 documentation. Openwisp-Radius.readthedocs.io. Retrieved August 3, 2023, from https://openwisp-radius.readthedocs.io/en/stable/user/enforcing_limits.html

OpenWISP. (n.d.-c). Values and Goals of OpenWISP — OpenWISP 22.05 documentation. Openwisp.io. Retrieved March 17, 2023, from https://openwisp.io/docs/general/values.html#goals

OpenWISP. (n.d.-d). WiFi Hotspot & Captive Portal — OpenWISP 22.05 documentation. Openwisp.io. Retrieved May 11, 2023, from https://openwisp.io/docs/tutorials/hotspot.html

openwisp. (2023a). NetJSON Templates. GitHub. https://github.com/openwisp/netjson-templates

openwisp. (2023b). openwisp-config. GitHub. https://github.com/openwisp/openwisp-config

openwisp. (2023c). openwisp-controller. GitHub. https://github.com/openwisp/openwisp-controller

openwisp. (2023d). openwisp-monitoring. GitHub. https://github.com/openwisp/openwisp-monitoring

openwisp. (2023e). openwisp-radius. GitHub. https://github.com/openwisp/openwisp-radius

United States Environmental Protection Agency. (2023). Lean Thinking and Methods - Kaizen. US EPA. https://www.epa.gov/sustainability/lean-thinking-and-methods-kaizen

Appendix B:

Technical Design Document: Module Marketplace for OpenWISP

Prepared by Alicia Pallarol Isábal as part of a Bachelor thesis submission to Universitat Politècnica de Catalunya in October 2023.

Index

A. Introduction **128**
B. Scope of the document **130**
C. Key features and functionalities of the module marketplace **131**
D. Constraints on the design **136**
E. System architecture and components **139**
F. Design decisions **141**
G. Open issues and challenges **145**
H. Conclusion **147**
H.1 Key Takeaways 147
H.2 Acknowledgement of Challenges 147
H.3 Next Steps 148
H.4 Closing Note 148
J. References **149**

List of Figures:

1. Module Marketplace architecture diagram 140

A. Introduction

OpenWISP, a widely recognised open-source network management system is gaining growth. Its ability to seamlessly manage and automate network configurations has paved the way for its expansive utility. Within this framework, the development of a module marketplace promises to offer users a centralised location to discover, acquire, and manage additional functionalities, improving their capacity to tailor the system to specific needs while keeping the collaboration and community principles of open-source software.

This Technical Design Document (TDD) has been composed as part of Alicia Pallarol Isábal's bachelor thesis in Computer Science, and it aims to lay a structured, technical foundation for developing the previously mentioned module marketplace. It intends to elucidate the proposed system architecture and components, design decisions, and other critical aspects integral to the successful realisation of the module.

Geared towards software developers, project managers, and other stakeholders involved in the OpenWISP project, this document offers comprehensive insights and directives, assuming a foundational understanding of network management principles and OpenWISP functionalities.

The purpose of the module marketplace is to fill a gap for network administrators using OpenWISP who have a specific need that would require a new module. These administrators could be able to find the modules they need just coded by other developers. It is also beneficial for developers to use this platform for distributing their modules, as it will help to promote collaboration and innovation within the OpenWISP community.

Embarking on this exploration, the document is segmented into various sections, including Scope, Key Features, Constraints, and more, each methodically unfolding the various facets

of the design and development process, through a roadmap to navigate through the intricate details of the marketplace module creation.

B. Scope of the document

The purpose of the module marketplace is to provide a platform for software developers to distribute and sell their modules. The target audience is network administrators using OpenWISP who have a specific need that would require a new module. The developers publishing the modules will do it using GitHub.

The key features and functionality of the module marketplace include:

1. A search engine to help developers find modules that meet their needs.
2. A user management system to identify module publishers.
3. A system for publishers to share their modules on landing pages.
4. An information landing page template.
5. A process for reviewing modules before landing pages are published.

The constraints on the design of the module marketplace include:

1. The module marketplace must be easy to use for both developers and users.
2. The module marketplace must be secure.
3. The module marketplace must be scalable to handle many modules and users.
4. The publishers' authentication needs to be integrated with GitHub.

C. Key features and functionalities of the module marketplace

In the previous section, we mentioned the key features of the marketplace. Here are these features with further details:

1. A search engine to help developers find modules that meet their needs: to assist developers in efficiently discovering modules that solve their requirements, a robust search engine will be embedded within the marketplace. This engine will leverage a multifaceted ranking algorithm, considering numerous variables to prioritize search results optimally:
    - *Relevance*: this will be determined through keyword matching and semantic understanding, employing algorithms and NLP (Natural Language Processing) techniques to correlate search queries with module descriptions and metadata.
    - *Popularity*: a module's popularity will be inferred from its GitHub traffic, and social signals, reflecting its acceptance and usage within the community.
    - *Rating*: integrating user ratings, review sentiments and additional quality indicators, the rating will serve as a direct reflection of user experiences and module robustness.
    - *Other factors*: furthermore, variables such as developer reputation, module recency, compatibility with OpenWISP, and any certification or verification will be significant in fine-tuning the search results, ensuring developers have access to relevant, quality modules.

2. A user management system to identify module publishers: central to the functionality of the module marketplace is the user management system, prepared to validate and identify module publishers through an integrative GitHub authentication system.
    - *Authentication and profile creation*: taking advantage of the reliability and universality of GitHub OAuth authentication, the system assures secure access and identity verification of module publishers, who are then prompted to augment their profiles with additional customisable information drawn from GitHub data.
    - *Module management*: empowering publishers to link directly to their GitHub repositories, the system ensures not only the consistency and recency of the hosted modules but also streamlined management of versioning, updates, and user notifications.
    - *Review, approval, and publication*: with an established review process, the platform maintains a qualitative standard, facilitating communication between reviewers and publishers, and ultimately culminating in the publication of vetted modules.
    - *Analytics, support, and interaction*: publishers are given a comprehensive dashboard, offering crucial insights into module statistics, while also establishing a conduit for user feedback, support, and documentation access.
    - *Security and compliance*: ensuring adherence to requisite legal, regulatory, and security norms, the system perpetually safeguards the marketplace's integrity, and by extension, its user base.
3. A system for publishers to share their modules on landing pages: for sharing modules, the marketplace will introduce a structured system, where publishers can create dedicated landing pages for their modules.

- *Landing page creation and customisation*: the system allows publishers to effortlessly input critical module information and customise their landing pages, fostering clarity and brand alignment while also providing a true-to-form preview before publication.
- *Visibility and accessibility*: with features like public/private modes and accessibility checks, it ensures that landing pages and modules are readily accessible and visible as per publisher preference.
- *User interaction and analytics*: by enabling user comments, ratings, and reviews, alongside providing critical analytics like view and download counts, the system ensures a bidirectional engagement between publishers and users.
- *Updates*: automated checks for repository updates and facilitating notifications ensure the module information stays current and relevant.
- *Module management*: options for deactivation, withdrawal, and data management guarantee publishers have complete control over the availability and presence of their modules and associated data.

4. An information landing page template: aiming to ensure a consistent, user-friendly, and accessible experience across all module landing pages, the module marketplace will introduce a standardised information landing page template.
    - *Uniformity and responsive design*: by having a predefined layout and ensuring responsiveness, the template establishes a uniform and accessible experience across diverse devices and user environments.
    - *Dynamic and automated content integration*: publishers are presented with clearly defined content fields, ensuring a balanced blend of uniformity and individuality, as their provided content is dynamically integrated into the strategic zones of the landing page.

- *Interactive elements and UX*: there should be interactive elements like download buttons and feedback sections to ensure user engagement is straightforward and rewarding, whilst also guaranteeing that navigation and user support are intuitively structured.

5. A process for reviewing modules before landing pages are published: to maintain high-quality, secure, and user-oriented modules, the marketplace will incorporate a thorough review process, ensuring that every module accords with the esteemed standards upheld by OpenWISP.
    - *Structured submission and verification*: publishers are facilitated with a structured submission form, the above-mentioned template, then, there will be an automated or manual acknowledgement to confirm the receipt of the module for review.
    - *Automated and manual reviews*: leaning on GitHub Actions, preliminary automated checks for aspects like code quality and dependencies will be performed, after those, OpenWISP developers will engage in a manual review with defined criteria.
    - *Constructive feedback and revision*: to facilitate transparency and improvement, detailed feedback will be given to publishers, who are then guided through a structured revision and re-submission process.
    - *Seamless approval to publishing transition*: upon approval, publishers are notified and, through an automated system, modules seamlessly transition to being published on the marketplace, adhering to the predetermined template.
    - *Continuous review of updates*: modules, after the initial approval, are subject to continuous review for any updates, ensuring that the marketplace consistently provides secure modules.

- *Uniform review with trained reviewers*: adherence to documented guidelines and periodic training for reviewers ensures that each module is evaluated under a uniform, fair, and knowledgeable perspective.

D. Constraints on the design

A more detailed description of the earlier mentioned constraints is this:

1. The module marketplace must be easy to use for both developers and users: to make the marketplace a relevant product, it needs to be usable and intuitive, for that, we need:
    - *Intuitive design*: crafting interfaces and workflows that are straightforward and self-explanatory for various user demographics.
    - *User assistance*: embedding help documentation, tooltips, and perhaps a chatbot to assist users in navigating and utilizing the marketplace.
    - *User feedback*: incorporating mechanisms like surveys, usability testing sessions, and feedback forms to extract insightful user feedback for continuous improvement.
    - *Accessibility*: ensuring that the platform is accessible to people with disabilities, adhering to Web Content Accessibility Guidelines (WCAG) to ensure that the platform is usable by people with varied abilities (W3C, n.d.).
    - *UX/UI design principles:* employing principles such as Fitts' Law, Hick's Law, and the Pareto Principle (uxtoast, n.d.) to enhance ease of use and efficiency.
2. The module marketplace must be secure: to be trustworthy to the users so that they freely use the marketplace it is essential to guarantee security:
    - *Data protection*: utilizing Advanced Encryption Standard (AES) for data encryption and ensuring secure data transmission through protocols such as TLS.
    - *Authentication and authorization*: Implementing secure authentication (integrating with GitHub for publishers) and ensuring appropriate authorization mechanisms.

- *Code and infrastructure security*: regularly auditing the codebase and infrastructure for vulnerabilities and addressing them promptly.
- *User privacy*: ensuring user privacy by collecting minimal data and providing transparent communication regarding data usage.

3. The module marketplace must be scalable to handle many modules and users: we need to ensure that the marketplace is a stable software solution.
    - *Infrastructure scalability*: designing infrastructure that can dynamically scale to accommodate growing data, user traffic, and module submissions using containerization (Docker) and orchestrators (Kubernetes) to ensure that the application infrastructure can dynamically scale.
    - *Database scalability*: employing a database system that ensures optimal performance and can manage increasing data without sacrificing speed or reliability by considering NoSQL databases like MongoDB which are known for their scalability and flexibility.
    - *Platform performance*: ensuring that the platform maintains its performance, including load times and responsiveness, even as the scale grows.
    - *Cost management*: implementing solutions that allow scalable infrastructure without causing costs to spiral out of control with cloud services (Amazon Web Services, Azure or Google Cloud) that provide automatic scaling capabilities to manage costs while scaling.
4. The publishers' authentication needs to be integrated with GitHub: in order to link easily the publisher with the module's repository it is important to be integrated with GitHub authentication:
    - *OAuth authentication*: employing GitHub OAuth for secure and straightforward publisher authentication, ensuring a secure and user-friendly authentication process, adhering to OAuth 2.0 specifications.

- *GitHub API integration*: ensuring reliable and secure interactions with the GitHub API for retrieving publisher data and repository details by implementing a queuing system to manage requests and adhere to GitHub's usage policies.
- *Data synchronization*: managing data synchronization between the marketplace and GitHub, ensuring consistency and reliability using webhooks or periodic API polling to maintain synchronization between data stored on the marketplace and GitHub repositories.
- *Error handling and user support*: implementing error handling for potential issues (like API failures) and providing support and guidance for publishers in such scenarios.

E. System architecture and components

It is time to discuss the architecture and the technologies. Figure 1 explains it visually but here is a description:

- *Frontend*: the chosen frontend technology is React due to speed, flexibility, and ease of use which will help to build complex user interfaces, such as the one for a module marketplace (Meta Open Source, 2023).
- *Backend*: the chosen technologies are node.js with express.js because of their efficiency, scalability and easy integration with NoSQL databases (OpenJS Foundation, 2017).
- *Search technology*: the chosen technology is Elasticsearch because of its full-text search capabilities and easy set-up and use. Moreover, it is scalable and can handle large amounts of data (Elastic, 2019).
- *CI/CD*: the chosen technology is GitHub actions because of the possibility to automate the building, testing, and deployment of the application. Also, it is easy to use and it is integrated with GitHub (GitHub, n.d.).
- *Orchestration and containerisation*: Kubernetes Engine and Docker are used to deploy and manage the backend application because they are very reliable and scalable. We will have the frontend, backend and search engine within our Kubernetes cluster (Campbell, n.d.) (Kubernetes, 2022).
- *Database*: as we discussed, we want to use a NoSQL database, specifically MongoDB is the chosen technology since it is document-oriented which we find useful if your modules have varying attributes. We want to locate the database outside the Kubernetes cluster to have more control over features related to backups, scaling and maintenance (MongoDB, 2019).

Using GitHub Actions for the CI/CD not only provides automation for building and testing the application but also smoothly integrates with the deployment process by pushing Docker images to the Docker Registry. Those images are then accessed by the Kubernetes cluster, which orchestrates the deployment of the various services, like the Node.js/Express.js backend and Elasticsearch, ensuring they communicate seamlessly and scale according to demand. Also, by having MongoDB externally, we ensure robust data management without overloading our application cluster.

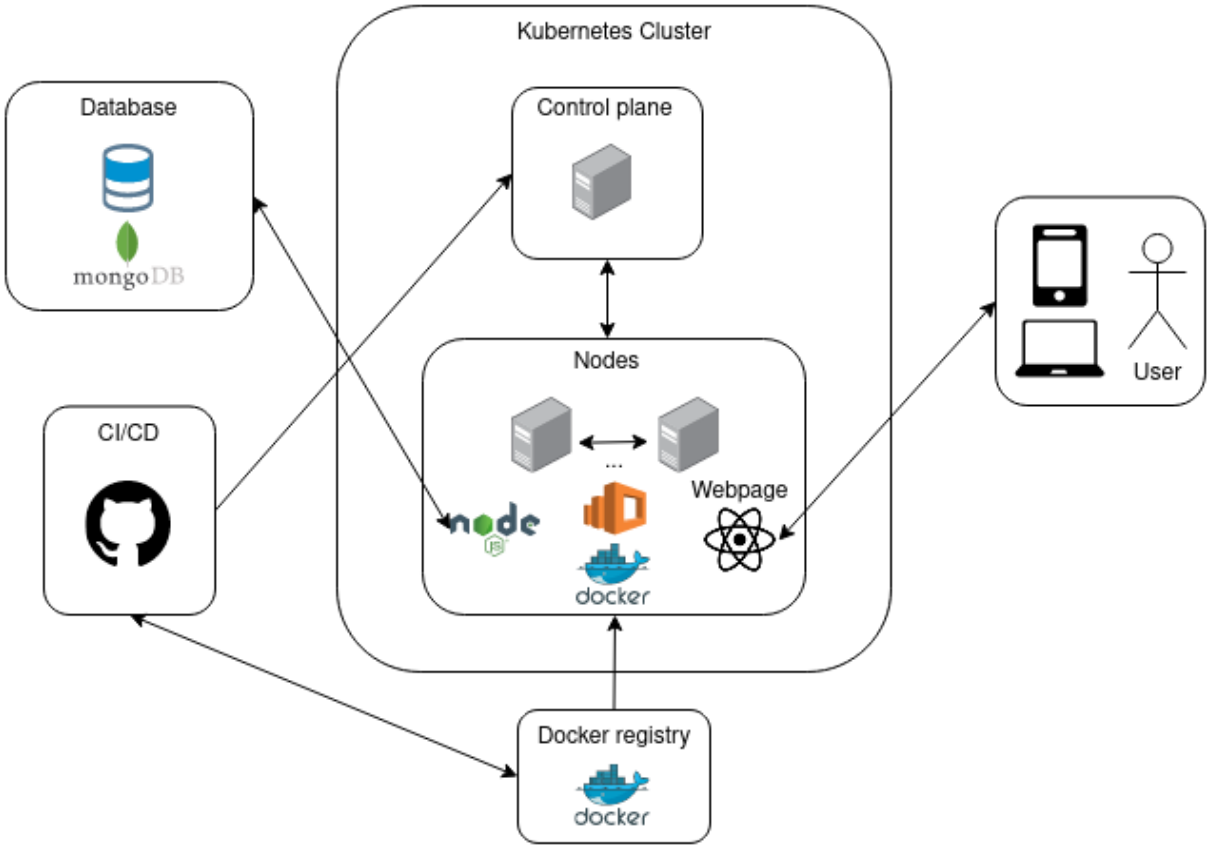

Figure 1[1]: Module Marketplace architecture diagram.

[1] Source: own elaboration using Draw.io according to the description at *chapter E*.

F. Design decisions

To develop the marketplace, some design decisions need to be made.

1. Technology choices:
    a. Rationale: Technologies like React, Node.js with Express.js, Elasticsearch, GitHub Actions, Docker, Kubernetes, and MongoDB were selected due to their widespread use, community support, scalability, and compatibility with each other.
    b. Implications: The chosen stack is modern and has a robust ecosystem, but also requires specific expertise for efficient development and management.
    c. Alternatives: While other technologies like Angular or SQL databases might have been viable, the chosen stack provided optimal solutions for our specific use cases, such as the need for a NoSQL database to handle varying module attributes.
2. Data management:
    a. Rationale: MongoDB, a NoSQL database, was chosen for its document-oriented nature and scalability.
    b. Implications: The use of MongoDB allows for flexibility in data structuring but implies the need for efficient data indexing and retrieval strategies to ensure performance.
    c. Alternatives: Other NoSQL databases, like CouchDB, were considered but MongoDB's rich feature set and broad usage in the industry were decisive factors.
3. UI/UX design:

   a. Rationale: A user-friendly, intuitive interface was prioritized to facilitate easy navigation and usage of the platform by both developers and network administrators.
   b. Implications: The focus on usability may necessitate ongoing UX research and adjustments based on user feedback and behaviour.
   c. Alternatives: Alternative design philosophies, like minimalist designs, were considered but ultimately, user-friendliness and providing ample information were deemed crucial.

4. Security protocols:
   a. Rationale: Implementing industry-standard security protocols to safeguard user data and ensure secure transactions.
   b. Implications: Rigorous testing and periodic security audits will be necessary to maintain high security.
   c. Alternatives: Different levels of security protocols were considered, but a robust approach was deemed essential due to the platform's nature.
5. Scalability and performance optimisation:
   a. Rationale: Kubernetes was chosen to ensure that the platform can scale efficiently to handle growing user and data loads.
   b. Implications: The use of Kubernetes necessitates effective monitoring to manage resources efficiently and ensure cost-effectiveness.
   c. Alternatives: Serverless architectures were considered but ruled out due to the complex state management and potential latency in certain use cases.
6. Integration and compatibility:
   a. Rationale: GitHub was integrated for user management and CI/CD to ensure familiarity for developers and streamline workflows.

   b. Implications: Dependency on GitHub APIs and ensuring ongoing compatibility with future GitHub updates.
   c. Alternatives: Other platforms or custom solutions were considered but GitHub was chosen for its widespread adoption among the target user base.

7. Development methodology (Hoory & Bottorff, 2022):
   a. Rationale: Agile, due to its iterative nature and flexibility to adapt to changes.
   b. Implications: Regular sprints and reviews will be integral to the development process and during this process, change will be possible.
   c. Alternatives: Waterfall methodology, discarded due to the lack of flexibility.
8. Testing and quality assurance:
   a. Rationale: A thorough QA and testing phase utilizing automated tests and manual review to ensure functionality and security.
   b. Implications: May require additional resources and time to ensure comprehensive testing.
   c. Alternatives: Different testing strategies and tools, but the chosen methods provide a balanced approach to quality and efficiency.
9. Development strategy:
   a. Rationale: Automated deployment through CI/CD pipelines to ensure consistent and error-free releases.
   b. Implications: Needs rigorous testing and well-defined deployment strategies to avoid issues in production.
   c. Alternatives: Manual deployment was considered but ruled out due to potential human error and inefficiency.
10. Maintenance and updates:
   a. Rationale: Regular updates and proactive maintenance to ensure security, functionality, and meeting user needs.

b. Implications: Necessitates effective communication channels with users to manage expectations during updates or maintenance periods.
c. Alternatives: Reactive approaches were considered, but proactive maintenance was deemed crucial for long-term sustainability and user trust.

G. Open issues and challenges

Some open issues include:

- Unpredictability in resource utilization:
    - Issue: This is a theoretical Design and there is no infrastructure to base the document, it is impossible to know or predict either the user number or the available resources.
    - Challenge: Designing the Kubernetes cluster, with the accurate resources that the control plane and the nodes have.
- User adoption and contribution:
    - Convincing developers and network administrators to use and contribute to the module marketplace.
    - Challenge: Developing engagement, trust, and a sense of community among users.
- Module quality and standardization:
    - Issue: Ensuring that the published modules adhere to quality and compatibility standards.
    - Challenge: Implementing a rigorous yet user-friendly review process to maintain a high-quality module inventory.
- Scalability and performance management:
    - Issue: Efficiently managing resources in Kubernetes to ensure cost-effectiveness and performance.
    - Challenge: Balancing between available resources and potential spikes in user activity or data loads.
- Legal and compliance aspects:

    - Issue: Managing the legal aspects of hosting code and modules, which might be used in various environments and applications.
    - Challenge: Navigating through licensing, intellectual property, and compliance, especially in an open-source context.
- Technical debt and future-proofing:
    - Issue: As technology evolves, ensuring that the platform adapts and remains relevant and efficient can be challenging.
    - Challenge: Managing technical debt and ensuring that the architecture and technologies used are future-proof to a reasonable extent.

H. Conclusion

The endeavour to create a Module Marketplace for OpenWISP presents a remarkable opportunity to facilitate network management processes by offering a centralised repository of reusable modules. This initiative, as outlined in the technical design, merges innovative technologies and practices, including Kubernetes, Docker, React, Node.js, Elasticsearch, and MongoDB, to build a scalable, user-friendly, and efficient platform.

H.1 Key Takeaways

- *Integrative approach*: The design emphasizes an integrative approach, amalgamating various technologies and practices to harness their collective capabilities and provide a robust platform.
- *Scalability and flexibility*: using Kubernetes and Docker ensures scalable and flexible architecture, accommodating growth and adapting to evolving requirements.
- *User-centric design*: With a strong focus on usability, the platform endeavours to provide a seamless and intuitive user experience for both module publishers and consumers.
- *Security and reliability*: While ensuring security through GitHub-based authentication, the infrastructure design maintains a consistent focus on reliability and data integrity.

H.2 Acknowledgement of Challenges

The project acknowledges several challenges, including managing scalability, ensuring consistent user engagement, maintaining module quality, and navigating through legal and compliance aspects.

Given the theoretical nature of this design, the actual implementation might uncover additional challenges and insights that will necessitate iterative enhancements to the system architecture and user interfaces.

H.3 Next Steps

- *Prototype development*: The initial phase would involve developing a Proof of Concept (PoC) to validate the technical design, gather early user feedback, and identify potential enhancement areas.
- *Continuous testing and feedback integration*: Implementing continuous testing and feedback loops to refine the platform and enhance its efficiency, usability, and reliability.
- *Community engagement*: Initiating programs and platforms to engage the developer community, encouraging contributions, and fostering a collaborative ecosystem.
- *Iterative development*: Adopting an agile and iterative development approach to manage changes, updates, and enhancements in an organized and risk-mitigated manner.

H.4 Closing Note

The module marketplace stands to be a central element in increasing the capabilities and efficiency of network administrators and developers interacting with OpenWISP. By providing a structured, efficient, and collaborative platform, the marketplace not only enhances module discoverability and reusability but also fosters a collaborative environment among developers and network administrators. As the project advances from this technical design to the development phase, continuous learning, adaptation, and community engagement will be critical in navigating towards a successful and sustainable module marketplace.

J. References

Campbell, J. (n.d.). Kubernetes vs. Docker. Atlassian. Retrieved August 9, 2023, from https://www.atlassian.com/microservices/microservices-architecture/kubernetes-vs-docker

Elastic. (2019). Open Source Search: The Creators of Elasticsearch, ELK Stack & Kibana | Elastic. Elastic.co; Elastic. https://www.elastic.co/

GitHub. (n.d.). Features • GitHub Actions. GitHub. Retrieved August 9, 2023, from https://github.com/features/actions

Hoory, L., & Bottorff, C. (2022, August 10). Agile vs. Waterfall: Which Project Management Methodology Should I Use? Forbes Advisor. https://www.forbes.com/advisor/business/agile-vs-waterfall-methodology/

Kubernetes. (2022, November 24). Communication between Nodes and the Control Plane. Kubernetes. https://kubernetes.io/docs/concepts/architecture/control-plane-node-communication/

Meta Open Source. (2023). React. React.dev. https://react.dev/

MongoDB. (2019). The most popular database for modern apps. MongoDB. https://www.mongodb.com

OpenJS Foundation. (2017). Express - Node.js web application framework. Expressjs.com. https://expressjs.com/

Uxtoast. (n.d.). uxtoast. Www.uxtoast.com. Retrieved August 9, 2023, from https://www.uxtoast.com/ux-laws/

W3C. (2018). Web Content Accessibility Guidelines (WCAG) Overview. Web Accessibility Initiative (WAI). https://www.w3.org/WAI/standards-guidelines/wcag/