

Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____Олексій СМІРНОВ
“ ____ ” _____2026 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за першим (бакалаврським) рівнем вищої освіти
на тему

**“ Програмне забезпечення системи моніторингу якості Wi-Fi-покриття
локальної мережі з використанням мережі сенсорів на базі ESP32”**

Виконав здобувач вищої освіти
IV курсу, групи КІ-22-2
ОПП «Комп’ютерна інженерія»
спеціальності 123 «Комп’ютерна інженерія»
_____Ткаченко О.Ю.
« ____ » _____2026 р.

Керівник проекту
доктор технічних наук, професор
_____Коваленко О.В.
« ____ » _____2026 р.
Рецензент _____

Центральноукраїнський національний технічний університет
Факультет Механіко-технологічний
Кафедра Кібербезпеки та програмного забезпечення
Освітній ступінь бакалавр
Галузь знань .12 "Інформаційні технології"
Спеціальність 123 "Комп'ютерна інженерія"
Освітньо-професійна (освітньо-наукова) програма "Комп'ютерна інженерія"

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____Олексій СМІРНОВ
« 07 » лютого 2026 року

**ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ
РОБОТУ ЗА ПЕРШИМ (БАКАЛАВРСЬКИМ) РІВНЕМ
ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ**

Ткаченку Олексію Юрійовичу

(прізвище, ім'я, по батькові)

1. Тема роботи *Програмне забезпечення системи моніторингу
якості Wi-Fi-покриття локальної мережі з
використанням мережі сенсорів на базі ESP32*

2. Керівник роботи *Коваленко Олександр Володимирович, докт. техн. наук,
професор*

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 136-02 від 27.02.2026 року

3. Строк подання студентом роботи до захисту 16.05.2026 р.

4. Мета та завдання випускної кваліфікаційної роботи: *Метою роботи є розробка
програмного забезпечення системи моніторингу якості Wi-Fi-покриття локальної
мережі з використанням мережі сенсорів на базі ESP32*

5. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно
розробити)

1. Призначення та область використання.

2. Перегляд аналогічних існуючих систем.

3. Опис і обґрунтування проектних рішень.

4. Етапи програмування системи.

5. Впровадження системи в промислову експлуатацію.

6. Висновки

6. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Структурна схема системи 1 аркуш

Функціональна схема системи 1 аркуш

Діаграма процесів 1 аркуш

Блок-схема алгоритму роботи додатку 2 аркуша

7. Дата видачі завдання « 07 » лютого 2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.03.2026 р.	
2.	Постановка задачі, оформлення ТЗ	15.03.2026 р.	
3.	Розробка моделі компонента	20.03.2026 р.	
4.	Розробка структур даних	25.03.2026 р.	
5.	Розробка алгоритмів зв'язку та відображення	30.03.2026 р.	
6.	Програмування алгоритмів	10.04.2026 р.	
7.	Оформлення ПЗ	17.04.2026 р.	
8.	Попередній захист роботи	16.05.2026 р.	

Дата видачі завдання
« 07 » лютого 2026 р.

Підпис керівника

_____ Коваленко О. В.
(прізвище та ініціали)

Завдання прийнято до виконання
« 07 » лютого 2026 р.

Підпис здобувача

_____ Ткаченко О. Ю.
(прізвище та ініціали)

АНОТАЦІЯ

Ткаченко О. Ю. Програмне забезпечення системи моніторингу якості Wi-Fi-покриття локальної мережі з використанням мережі сенсорів на базі ESP32. 123 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2026.

В даній бакалаврській роботі розроблено програмне забезпечення системи моніторингу якості Wi-Fi покриття локальної мережі з використанням мережі сенсорних вузлів на базі мікроконтролерів ESP32.

Метою розробки є створення програмної системи, що забезпечує збір, обробку та візуалізацію даних про рівень сигналу бездротової мережі для оцінки якості радіопокриття.

Результатом роботи є реалізована система, яка дозволяє отримувати показники рівня сигналу Wi-Fi від сенсорних вузлів, передавати їх до центрального програмного модуля та відображати результати у вигляді карти покриття.

В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

Розроблено зручний інтерфейс користувача. Наведені інструкції по роботі з програмними засобами.

Програма може використовуватися на персональних комп'ютерах під керуванням операційних систем Windows та Linux.

Програму розроблено в середовищах Qt Creator та Arduino IDE із використанням мови програмування C++.

Ключові слова: комп'ютерна інженерія, Wi-Fi-покриття, моніторинг якості сигналу, RSSI, ESP32, бездротова сенсорна мережа, IoT, MQTT

ABSTRACT

Tkachenko O. Yu. Software for Monitoring the Quality of Wi-Fi Coverage in a Local Network Using a Sensor Network Based on ESP32. 123 Computer Engineering. Central Ukrainian National Technical University. Kropyvnytskyi. 2026.

In this bachelor's thesis, software for monitoring the quality of Wi-Fi coverage in a local network using a sensor network based on ESP32 microcontrollers has been developed.

The purpose of the development is to create a software system that provides collection, processing, and visualization of wireless network signal strength data for evaluating the quality of radio coverage.

The result of the work is an implemented system that allows obtaining Wi-Fi signal strength indicators from sensor nodes, transmitting them to a central software module, and displaying the results in the form of a coverage map.

During the development of the software model, an analysis of existing hardware and software tools was carried out. All components of the developed software are fully described.

A user-friendly interface has been developed. Instructions for working with the software tools are provided.

The program can be used on personal computers running the Windows and Linux operating systems.

The program was developed using Qt Creator and Arduino IDE with the C++ programming language.

Keywords: computer engineering, Wi-Fi coverage, signal quality monitoring, RSSI, ESP32, wireless sensor network, IoT, MQTT

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ	2
ВСТУП.....	3
1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ	5
1.1 Призначення системи.....	5
1.2 Область застосування	6
2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ	8
2.1 Огляд існуючих систем, технологій, архітектур та програмних рішень за профілем теми випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	8
2.2 Обґрунтування вибору засобів для побудови системи та мови програмування	16
2.3 Розгорнута постановка завдання	17
3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ	19
3.1 Опис функціонування системи	19
3.2 Розробка структурної схеми.....	23
3.3 Розробка функціональної схеми	25
3.4 Розробка діаграми процесів	27
4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ	30
4.1 Розробка блок-схем та опис алгоритмів функціонування системи	30
4.2 Захист розробленого програмного забезпечення.....	43
5 ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ	46
6 ОСНОВНІ ВИСНОВКИ.....	50
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	52

					<i>ВКРБ-123.26.0060.00.00.ПЗ</i>			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розробив		Ткаченко О. Ю.			Програмне забезпечення системи моніторингу якості Wi-Fi-покриття локальної мережі з використанням мережі сенсорів на базі ESP32	Літ.	Арк.	Акрушів
Перевірів		Коваленко О. В.				Б	1	58
						ЦНТУ КІ-22-2		
Н. контр.		Коваленко А. С						
Затв.		Смірнов О. А.						

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

AES	–	симетричний алгоритм шифрування даних
BLE	–	Bluetooth з низьким енергоспоживанням
dBm	–	одиниця вимірювання потужності сигналу
ECDH	–	протокол обміну ключами
ESP32	–	мікроконтролер з вбудованими модулями
HTTP	–	протокол передачі гіпертексту
JSON	–	текстовий формат обміну даними
mbedTLS	–	бібліотека криптографічних примітивів
MQTT	–	легкий протокол обміну повідомленнями
Qt	–	кросплатформний фреймворк для розробки
RSSI	–	показник рівня потужності прийнятого сигналу
SSID	–	назва бездротової Wi-Fi мережі
UUID	–	універсальний унікальний ідентифікатор
Wi-Fi	–	стандарт бездротового локального зв'язку

ВСТУП

Актуальність теми: Якість Wi-Fi покриття у приміщенні залежить від планування, матеріалів стін, розташування точки доступу і активності сусідніх мереж. Оцінити їх вплив без систематичних вимірювань складно. Існуючі підходи зазвичай передбачають ручний обхід приміщення з пристроєм, що фіксує рівень сигналу. Такий метод дає лише миттєвий знімок і не відображає поведінку мережі в часі.

ESP32 має вбудований Wi-Fi і BLE модулі та низьку вартість. Я використав три таких вузли як мобільні вимірювальні точки: вузли послідовно переміщуються між позиціями у приміщенні, вимірюють RSSI від точки доступу і між собою через BLE. Щоб розмістити виміряні значення на просторовій карті без ручної фіксації координат, я реалізував трилатерацію, яка оцінює відносне положення вузлів на основі зібраних RSSI і BLE даних. Канал первинного налаштування вузлів захищено через BLE з ECDH обміном ключами і AES шифруванням переданих даних.

Мета й завдання дослідження: розробити програмне забезпечення системи моніторингу якості Wi-Fi покриття локальної мережі на основі мобільних сенсорних вузлів ESP32.

Для досягнення мети я визначив такі завдання: проаналізувати наявні підходи до моніторингу Wi-Fi покриття, обґрунтувати вибір RSSI як основної метрики, розробити прошивку для вузлів на базі ESP32 з захищеним BLE провізійонуванням, розробити desktop-застосунок для прийому даних через MQTT і їх візуалізації, реалізувати трилатерацію для автоматичного визначення позицій вузлів на карті, протестувати систему з трьома вузлами в умовах реальної локальної мережі.

Практична цінність: система дозволяє обстежити приміщення, послідовно переміщуючи вузли між точками вимірювання, і отримати карту

Зми	Анк	№ докум	Підпис	Дата

ВКРБ-123.26.0060.00.00.ПЗ

Арк.

3

рівня сигналу без ручної прив'язки координат вузлів. Трилатерація на основі RSSI і BLE автоматично розміщує кожен вимір на карті. Тестування підтвердило можливість виявлення ділянок з помітним просіданням сигналу. Апаратна база на ESP32 не потребує спеціалізованого обладнання.

К6ПЗ_2026

Змн	Арк	№ докум	Підпис	Дата

ВКРБ-123.26.0060.00.00.ПЗ

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Система призначена для вимірювання рівня Wi-Fi сигналу у приміщенні з метою виявлення ділянок з недостатнім покриттям. Вона складається з трьох мобільних вимірювальних вузлів на базі ESP32, які передають дані на десктопний додаток розроблений засобами Qt, де ці дані відображаються у вигляді теплової карти.

Кожен вузол є автономним пристроєм на базі мікроконтролера ESP32 з вбудованим Wi-Fi і BLE модулями. Прошивка написана мовою C++ у середовищі Arduino IDE. Ініціалізація реалізована у два етапи. Майстер-вузол піднімає власну точку доступу, користувач підключається до неї і через вбудований веб-інтерфейс вводить параметри локальної мережі і MQTT. Після цього майстер починає BLE рекламування і два вузли-ноди підключаються до нього по черзі, отримують ті самі параметри через зашифрований BLE канал. Шифрування реалізовано через ECDH обмін ключами і AES-128, тобто параметри мережі передаються між пристроями у зашифрованому вигляді і у відкритому ефірі не з'являються. Після ініціалізації всі три вузли підключаються до локальної мережі і готові до роботи без перепрошивки.

Передача даних між вузлами і додатком здійснюється за протоколом MQTT. Кожен вузол публікує виміряні значення RSSI від точки доступу і RSSI від сусідніх вузлів через BLE в один спільний топик. Додаток підписаний на цей топик і отримує дані в реальному часі. Система працює повністю в межах локальної мережі без окремого хмарного сервера.

Десктопний додаток написаний мовою C++ з використанням Qt. Він приймає MQTT повідомлення від вузлів і відображає значення RSSI у вигляді теплової карти. Щоб розмістити кожен вимір на карті без ручної прив'язки

Зми	Арк	№ докум	Підпис	Дата

координат, я реалізував трилатерацію на основі RSSI від точки доступу і BLE відстаней між вузлами. Алгоритм оцінює відносне положення кожного вузла на сітці і оновлює карту після кожного циклу вимірювань. Дані сесії записуються у CSV файл з відміткою часу, координатами вузлів і згладженими значеннями RSSI.

Вузли не прив'язані до фіксованих місць. Їх послідовно переміщують між точками приміщення, і теплова карта оновлюється в міру надходження нових вимірювань. Типовий сценарій виглядає так: користувач ініціалізує вузли через майстер, розміщує їх у різних частинах приміщення, запускає додаток, ставить точку доступу на тепловій карті і спостерігає за формуванням карти покриття. На основі отриманої картини можна визначити проблемні зони і прийняти рішення щодо розташування точки доступу.

1.2 Область застосування

Система розрахована на використання у невеликих приміщеннях з однією точкою доступу, де потрібно зрозуміти як розподіляється сигнал по площі. Я розробляв і тестував її у квартирі, тому саме цей сценарій є найбільш перевіреним.

Практично це означає будь-яке житлове або невелике робоче приміщення, де є локальна Wi-Fi мережа і є підозра що покриття нерівномірне. Наприклад, роутер стоїть в одній кімнаті, а в іншому кінці квартири з'єднання нестабільне, але незрозуміло наскільки саме і де починається просідання сигналу. Три вузли ESP32 розставляються по приміщенню, і додаток показує теплову карту на основі поточних вимірювань.

Десктопний додаток компілюється під Windows, Linux і macOS засобами Qt без змін у коді. Практичне тестування проводилось на Windows, тому можу стверджувати підтверджену працездатність саме на цій платформі. Збірка під інші операційні системи не перевірялась в рамках цієї роботи.

Зми	Арк	№ докум	Підпис	Дата

Щодо вимог до мережі і налаштування: перед початком роботи користувач підключається до точки доступу яку піднімає майстер-вузол, і через веб-інтерфейс вводить параметри локальної мережі. Після цього всі три вузли підключаються до локальної мережі. Додаткова вимога, наявність MQTT-брокера у цій мережі, до якого підключаються і вузли і додаток одночасно. Зовнішніх сервісів система не потребує.

Система не розрахована на великі об'єкти з кількома точками доступу або на приміщення де потрібна висока точність позиціонування. Три вузли дають три точки вимірювання одночасно, що достатньо для загальної оцінки покриття невеликого приміщення.

К6ПЗ-2026

Зми	Арк	№ докум	Підпис	Дата

ВКРБ-123.26.0060.00.00.ПЗ

Арк.

7

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

2.1 Огляд існуючих систем, технологій, архітектур та програмних рішень за профілем теми випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти

Для визначення місця розробленої системи серед існуючих рішень я розглянув три категорії: комерційне програмне забезпечення для обстеження Wi-Fi покриття, підходи до побудови мереж сенсорів на базі мікроконтролерів та протоколи передачі даних у подібних системах.

NetSpot

NetSpot є програмним інструментом для обстеження безпроводових мереж, аналізу Wi-Fi покриття та продуктивності. Він підтримує стандарти 802.11a/b/g/n/ac/ax і використовує стандартний Wi-Fi адаптер пристрою для вимірювання рівня сигналу та інших параметрів мережі з подальшою побудовою звітів.

Принцип роботи NetSpot полягає в тому, що користувач вказує своє поточне місцезнаходження на карті приміщення, після чого додаток починає вимірювати рівень сигналу. Далі користувач переміщується по приміщенню, фіксуючи вимірювання в кожній точці яку хоче проаналізувати. Цей підхід вимагає постійної присутності людини з ноутбуком або смартфоном під час обстеження і не дозволяє збирати дані з кількох точок одночасно.

Програма пропонує два основних режими роботи. Перший, Discover, сканує оточення і відображає всі видимі мережі, їх канали, рівень сигналу в dBm і відношення сигнал/шум. Другий, Survey, є основним режимом для побудови теплових карт, де користувач завантажує план приміщення або малює його вручну, а потім крок за кроком проставляє точки вимірювання. Кожна точка прив'язується до координат на плані вручну, натисканням миші

Зми	Арк	№ докум	Підпис	Дата

або дотиком. Автоматичного позиціонування немає, точність карти повністю залежить від уважності користувача під час обходу.

NetSpot пропонує понад 20 варіантів теплових карт і надає можливість аналізувати всі точки доступу в зоні досяжності. Кожна зона проекту може мати кілька знімків, зроблених у різний час доби або у різні дні з різними конфігураціями, що дозволяє порівнювати результати і визначати що працює найкраще в конкретному середовищі. Серед параметрів, які відображаються на картах, є рівень сигналу (RSSI), відношення сигнал/шум (SNR), перекриття каналів, зона покриття кожної точки доступу окремо і рівень завантаженості частотного діапазону. На рисунку 2.1 наведено головне вікно програми з прикладом теплової карти.

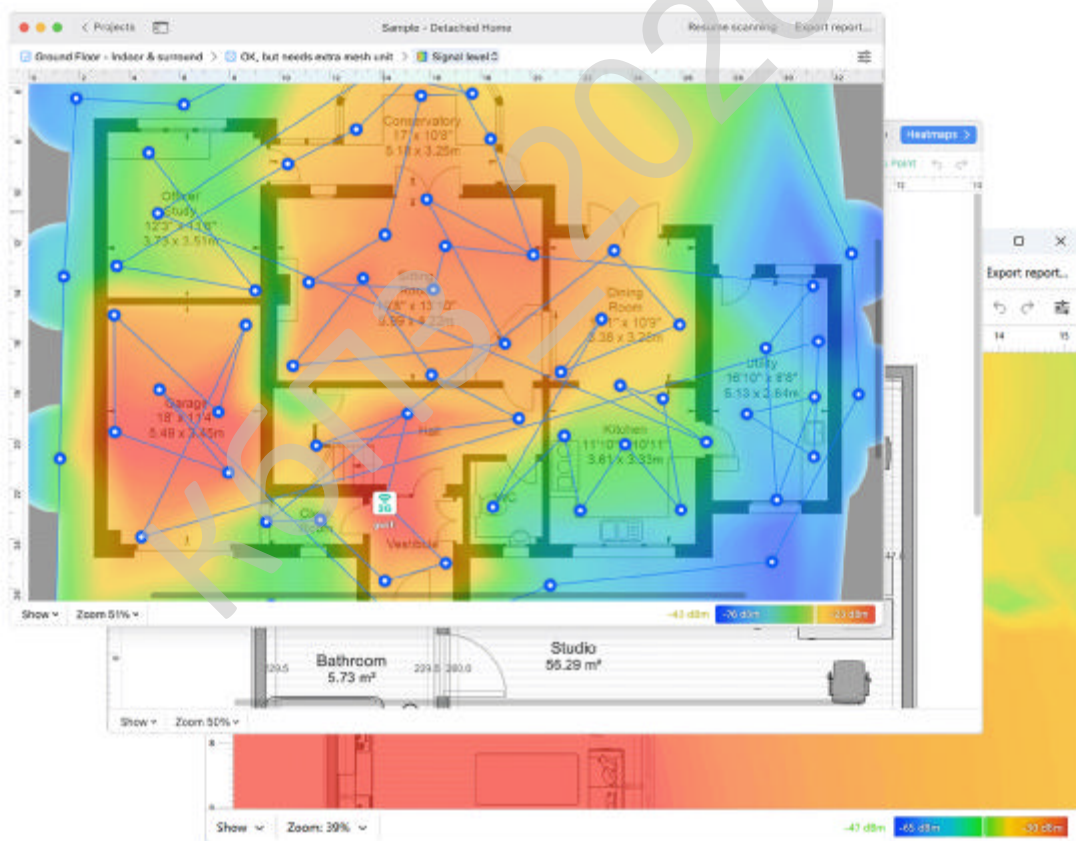


Рисунок 2.1 – Інтерфейс програми NetSpot

Програма доступна у трьох варіантах. Безкоштовна версія Home обмежена двома точками сканування на проект, що робить її придатною хіба

Змн	Арк	№ докум	Підпис	Дата

для швидкої перевірки в одній кімнаті. Версія PRO коштує близько 99 доларів США і знімає обмеження на кількість точок. Версія для організацій коштує близько 149 доларів США і додає функції для командної роботи та розширений експорт звітів. Ліцензія прив'язана до конкретного пристрою, а не до облікового запису.

З точки зору архітектури NetSpot є централізованим односенсорним рішенням: всі вимірювання виконує один пристрій, яким керує користувач вручну. Програмне забезпечення підтримує macOS і Windows, а також Android і iOS для спрощеного збору даних. Дані зберігаються локально у вигляді проектів і не потребують підключення до інтернету.

Є одна особливість, яку варто врахувати. Вбудований Wi-Fi адаптер більшості ноутбуків не переходить у повноцінний режим моніторингу так само, як спеціалізовані адаптери. Через це NetSpot бачить тільки ті мережі, які активно транслюють свій SSID, і може пропускати приховані мережі або пристрої, що не передають beacon-фреймів у момент сканування. На практиці для домашнього або офісного використання це не критично, але треба розуміти це обмеження.

Порівняно з розробленою системою NetSpot орієнтований на детальне ручне обстеження і надає значно більше параметрів для аналізу. Водночас він принципово відрізняється підходом до збору даних: один пристрій в руках користувача проти трьох автономних вузлів, які передають дані без участі людини.

Acrylic WiFi Heatmaps

Acrylic WiFi Heatmaps є інструментом для планування Wi-Fi мереж і проведення обстежень покриття. Він дозволяє проектувати, аналізувати та виявляти проблеми Wi-Fi, а також генерувати звіти. Інструмент підтримує всі стандарти Wi-Fi включно з Wi-Fi 7 і працює на частотах 2.4, 5 та 6 ГГц.

Зми	Арк	№ докум	Підпис	Дата

ВКРБ-123.26.0060.00.00.ПЗ

Арк.

10

Acrylic WiFi Heatmaps працює виключно під Windows і не підтримує macOS або інші платформи. Це практичне обмеження для користувачів, які працюють у змішаному середовищі.

Процес обстеження схожий на NetSpot: завантажуєте план приміщення, обходите його з ноутбуком і вручну проставляєте точки вимірювання. Відмінність полягає у режимі активного обстеження. У цьому режимі програма не просто вимірює рівень сигналу, а й тестує реальну пропускну здатність і затримку з'єднання до конкретної точки доступу. Для цього потрібні два Wi-Fi адаптери одночасно: один підключений до мережі для отримання даних про затримку і пропускну здатність, а другий у режимі моніторингу для збору інформації про інші мережі в зоні досяжності. На більшості ноутбуків є лише один вбудований адаптер, тому для повноцінної роботи потрібен додатковий USB-адаптер з підтримкою режиму моніторингу.

У режимі моніторингу Acrylic WiFi Heatmaps здатний відображати всі клієнтські пристрої, підключені до точок доступу. Це корисно, коли потрібно не просто оцінити покриття, а зрозуміти, які пристрої де знаходяться і з яким рівнем сигналу вони підключені. Для домашнього використання ця функція надлишкова, але у корпоративному середовищі вона дозволяє виявити несанкціоновані пристрої або проблемні клієнти, що погіршують якість мережі.

Для побудови теплових карт програма використовує алгоритми інтерполяції даних і генерує до 14 різних типів карт. Серед них є карти покриття RSSI, карти відношення сигнал/шум, карти перекриття каналів, карти пропускну здатності і карти затримки. Останні два типи доступні тільки після активного обстеження з двома адаптерами. Карты можна експортувати для перегляду в Google Earth або Google Maps у форматах KMZ і KML, а також у звіти у форматі DOCX.

Вартість Acrylic WiFi Heatmaps складає близько 199 євро за ліцензію для одного пристрою. Існує також версія для команд з ліцензуванням за кількістю

Зми	Анк	№ докум	Підпис	Дата

ВКРБ-123.26.0060.00.00.ПЗ

Арк.

11

користувачів. Безкоштовна пробна версія обмежена за кількістю точок сканування і не дозволяє повноцінно оцінити великі приміщення.

Один момент, який я вважаю важливим: Acrylic WiFi Heatmaps чітко розрахований на мережевих інженерів. Інтерфейс перевантажений технічними параметрами і вимагає розуміння того, що таке BSSID, beacon interval або spatial streams. На рисунку 2.2 наведений інтерфейс програми з тепловою картою. Для разового обстеження невеликого офісу або квартири цей рівень деталізації не потрібен і тільки ускладнює роботу.

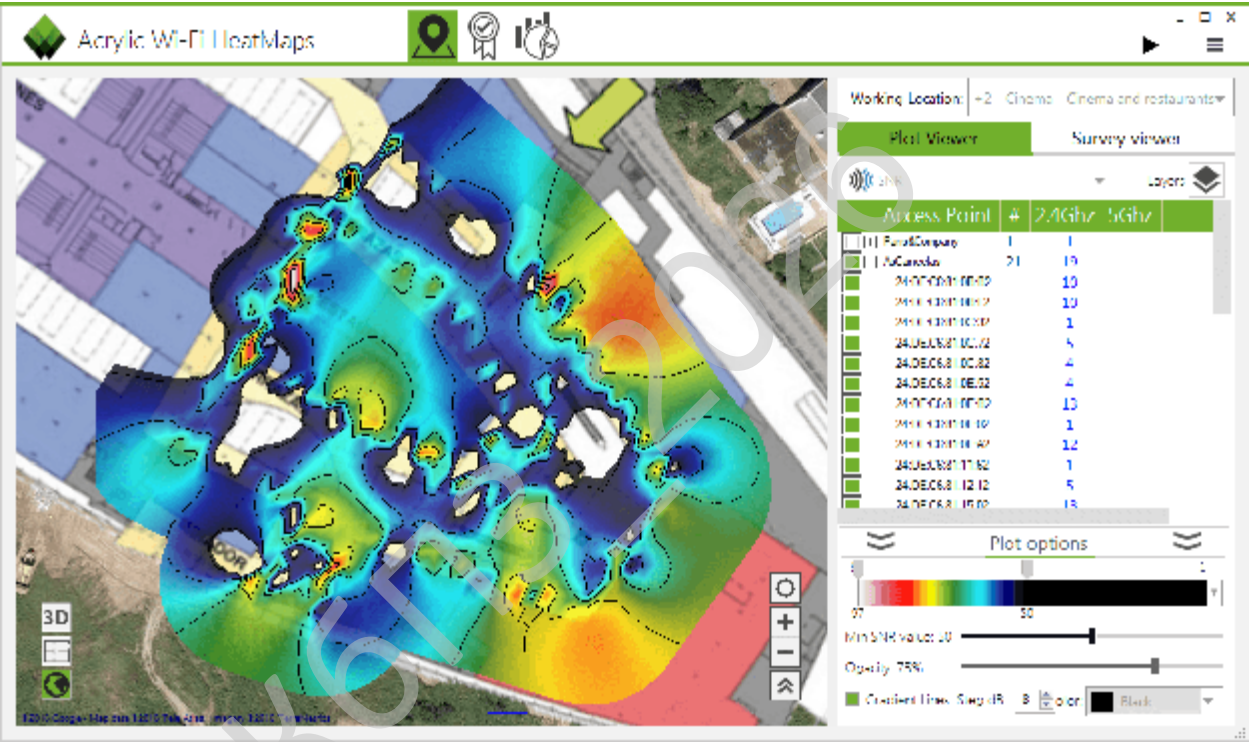


Рисунок 2.2 – Інтерфейс програми Acrylic Wi-Fi HeatMaps

Acrylic WiFi Heatmaps є значно складнішим інструментом ніж розроблена система і орієнтований передусім на мережевих інженерів та корпоративних користувачів. Він потребує спеціалізованого обладнання для роботи у режимі моніторингу і не передбачає автономних вимірювальних вузлів.

Ekahau AI Pro

Ekahau є одним з провідних інструментів для планування, аналізу та усунення несправностей Wi-Fi мереж, яким користуються 40% компаній зі списку Fortune 500. Це визначає і його цільову аудиторію: великі організації з розгалуженою мережевою інфраструктурою.

Ekahau AI Pro працює на macOS і Windows і використовує спеціалізований апаратний пристрій Ekahau Sidekick 2 з двома Wi-Fi радіомодулями для одночасного обстеження всіх Wi-Fi мереж включно з Wi-Fi 6E. Sidekick 2 підключається до ноутбука через USB і бере на себе всі радіовимірювання, звільняючи вбудований адаптер ноутбука для звичайної роботи. Пристрій також має вбудований GPS-приймач для роботи на відкритих майданчиках і сенсор температури. Вага Sidekick 2 складає близько 450 грамів, тобто під час обстеження великого приміщення ви носите з собою ноутбук і цей додатковий блок.

Процес роботи з Ekahau AI Pro починається з імпорту планів будівлі. Програма підтримує PDF, DXF і звичайні растрові формати. Далі мережевий інженер вносить дані про матеріали стін, стель і підлог, після чого програма будує прогнозовану модель покриття ще до початку фізичного обстеження. Це дозволяє одразу виявити потенційно проблемні зони і спланувати розміщення точок доступу до їх фактичного встановлення. На рисунку 2.3 наведено інтерфейс програми Ekahau AI Pro з прикладом прогнозованої моделі покриття. Після монтажу мережі проводиться фактичне обстеження, яке порівнюється з прогнозом.

Зми	Арх	№ докум	Підпис	Дата

ВКРБ-123.26.0060.00.00.ПЗ

Арк.

13

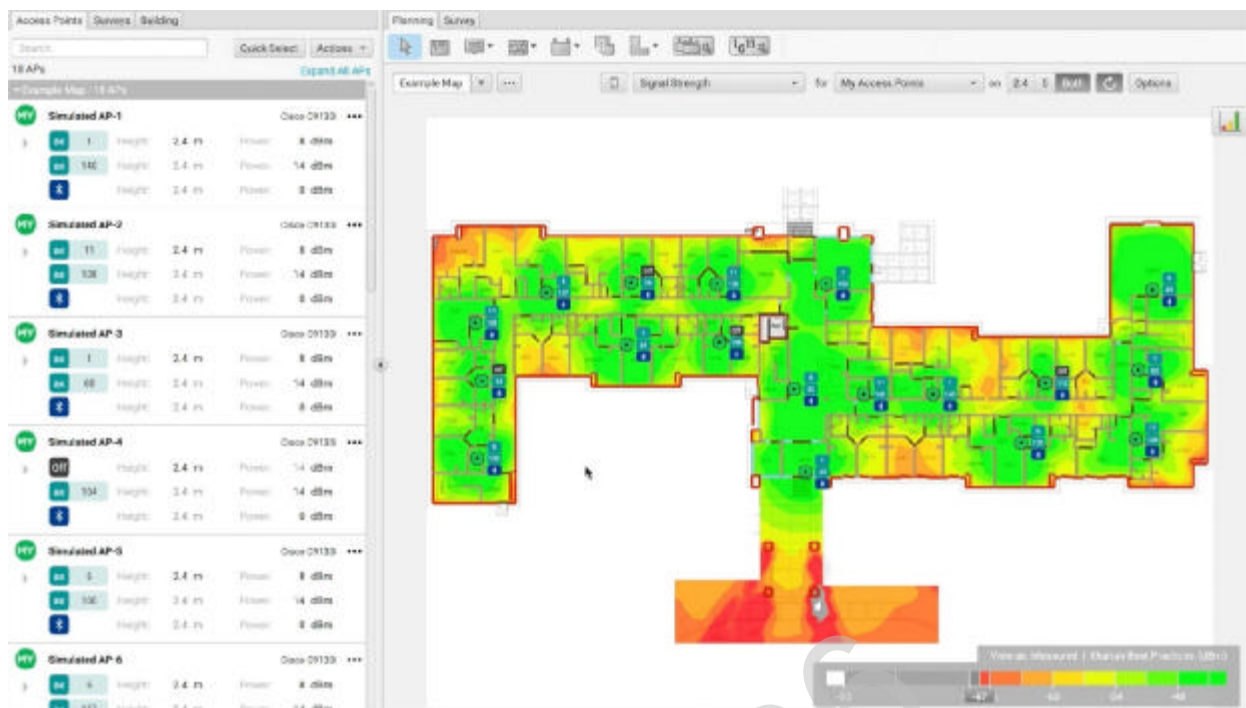


Рисунок 2.3 – Інтерфейс програми Ekahau AI Pro

Ekahau AI Pro підтримує одночасне проведення пасивних, активних і спектральних обстежень за один обхід приміщення. Пасивне обстеження вимірює рівень сигналу від усіх точок доступу в зоні видимості. Активне тестує реальну пропускну здатність і затримку. Спектральний аналіз виявляє джерела радіозавад, які можуть деградувати якість мережі навіть при достатньому рівні сигналу. Мікрохвильові печі, Bluetooth-пристрої, бездротові камери і деякі промислові пристрої генерують завади у діапазоні 2.4 ГГц, і Ekahau показує це на окремій карті.

Результати синхронізуються з хмарним сховищем Ekahau Cloud і доступні для спільної роботи кількох спеціалістів. Це важливо для великих проектів, де обстеження проводить одна команда, а проектування мережі виконує інша.

Вартість Ekahau AI Pro разом з обов'язковою річною підпискою починається від 7990 доларів США. Сюди входить і сам Sidekick 2, вартість якого окремо складає близько 1999 доларів США. Підписку треба продовжувати щороку, інакше програма переходить у режим обмеженого

Змн	Апк	№ докум	Підпис	Дата

ВКРБ-123.26.0060.00.00.ПЗ

функціоналу. Для компанії, яка регулярно проводить мережеві обстеження, ця вартість виправдана. Для разового проекту або невеликого приміщення, це нераціональні витрати.

На основі зібраних даних програма будує тривимірні плани приміщень з урахуванням матеріалів стін і підлог. Функція AI у назві стосується автоматичного розміщення точок доступу: програма аналізує результати обстеження і пропонує оптимальні позиції та налаштування для нових точок доступу з урахуванням стандартів покриття та перекриття сигналу.

Для оцінки покриття в невеликому приміщенні з одним роутером такий інструмент є надлишковим як за функціональністю, так і за ціною. Саме це і обґрунтовує доцільність апаратної бази на ESP32 як альтернативи для некорпоративного використання.

Порівняння з розробленою системою

Розглянуті комерційні інструменти об'єднує спільний підхід: один пристрій в руках користувача обходить приміщення і збирає дані послідовно. Розроблена система використовує три вузли, які розставляються у різних точках приміщення одночасно, що дозволяє отримувати паралельні вимірювання з кількох місць за один раз. Після збору даних вузли переносяться в наступні точки.

Є ще одна відмінність, яку варто зазначити. NetSpot, Acrylic і Ekahau потребують ноутбука або спеціалізованого обладнання вартістю від кількох сотень до кількох тисяч доларів. Вартість одного вузла розробленої системи на базі ESP32 з необхідними компонентами не перевищує 10 доларів США. Це не означає, що розроблена система краща, вона вирішує вужче завдання і надає менше параметрів. Але для оцінки покриття в домашньому або невеликому офісному середовищі різниця у вартості та складності є суттєвим аргументом.

Жоден з розглянутих комерційних інструментів не передбачає кількох одночасних точок вимірювання на базі апаратних вузлів, що є основною архітектурною відмінністю розробленої системи.

Зми	Арк	№ докум	Підпис	Дата

2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

ESP32 як апаратна база

Вибір ESP32 визначився тим, що цей мікроконтролер має вбудований Wi-Fi і BLE в одному чіпі. Це принципово для даної системи: Wi-Fi потрібен для вимірювання RSSI і передачі даних через MQTT, а BLE використовується для провізійнування вузлів і для вимірювання відстані між ними через RSSI. Альтернатива у вигляді ESP8266 має лише Wi-Fi без BLE, що унеможливило б реалізацію BLE-провізійнування і міжвузлових вимірювань. Arduino з зовнішнім Wi-Fi модулем теоретично можливий, але це окремий адаптер, окремі бібліотеки і значно більша складність схемотехніки без жодних переваг. ESP32 закриває обидві потреби одним модулем за низьку ціну, що і визначило вибір.

Arduino IDE та мова C++

Прошивка написана в Arduino IDE мовою C++. Реальна альтернатива тут це ESP-IDF, офіційний фреймворк від Espressif, який дає більш прямий доступ до апаратури і більше контролю. Я обрав Arduino IDE через готову підтримку ESP32 через менеджер плат і наявність перевірених бібліотек, зокрема PubSubClient для MQTT, WiFi.h для підключення і BLEDevice для роботи з BLE. ESP-IDF вимагав би більше часу на налаштування і написання обгорток для тих самих функцій. Для задач цього проекту різниця у продуктивності між двома підходами не є критичною, тому Arduino IDE виявився достатнім вибором.

Протокол MQTT

Для передачі даних між вузлами і десктопним додатком я розглядав два варіанти: HTTP і MQTT. HTTP передбачає що клієнт сам опитує сервер, тобто додаток мав би постійно робити запити до кожного вузла окремо. Це ускладнює архітектуру і створює затримки. MQTT працює інакше: вузли

Змн	Арк	№ докум	Підпис	Дата

публікують дані в топик, додаток підписаний на цей топик і отримує дані одразу після публікації. Три вузли публікують незалежно, додаток отримує все через одне з'єднання з брокером. Мінімальний розмір керуючого повідомлення MQTT становить два байти, що підходить для ресурсообмеженого ESP32. Брокер працює локально в межах мережі без залежності від зовнішніх сервісів.

Qt та C++ для десктопного додатку

Десктопний додаток написаний на C++ з використанням Qt. Я розглядав варіант веб-інтерфейсу на Python, але Qt мав конкретну перевагу: вбудований модуль Qt MQTT який підключається до брокера без сторонніх залежностей. Крім того, Qt дає інструменти для побудови власного віджета теплової карти з повним контролем над відображенням, що було потрібно для реалізації трилатерації і накладання плану будівлі. Використання C++ і для прошивки і для додатку скоротило кількість мов у проекті до однієї, що спростило підтримку коду в цілому.

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускню кваліфікаційну роботу за першим (бакалаврським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи моніторингу якості Wi-Fi-покриття локальної мережі з використанням мережі сенсорів на базі ESP32.

В процесі розробки випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти необхідно виконати наступний обсяг роботи:

а) провести аналіз існуючих систем-аналогів для виявлення їх позитивних і негативних якостей. Результати аналізу врахувати в подальших розробках;

б) вибрати та обґрунтувати методику побудови системи моніторингу якості Wi-Fi-покриття на основі мережі сенсорів ESP32. Розробити функціональну та структурну схеми системи;

в) розробити програмне забезпечення системи, що дозволить реалізувати поставлену технічним завданням задачу. Побудувати блок-схеми алгоритмів основної програми та підпрограм;

г) організувати інтерфейс користувача з метою відображення теплової карти покриття Wi-Fi сигналу, налаштування параметрів моделі розповсюдження сигналу та виводу на екран повідомлень про некоректні дії користувача і нестандартні ситуації в роботі системи;

д) розробити рекомендації по організаційних та методичних заходах, які забезпечать впровадження системи моніторингу Wi-Fi-покриття в експлуатацію та її подальше успішне використання;

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

ж) розробити заходи по охороні праці при впровадженні та експлуатації системи, а також розробити заходи з цивільного захисту;

з) сформулювати висновки про виконаний обсяг робіт та одержані результати.

Зми	Арк	№ докум	Підпис	Дата

ВКРБ-123.26.0060.00.00.ПЗ

Арк.

18

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

Система складається з трьох апаратних вузлів на базі ESP32 і десктопного додатка. Один вузол є головним, два інших підлеглими. Кожен вузол вимірює рівень сигналу Wi-Fi точки доступу і рівень BLE-сигналу від сусідніх вузлів, після чого передає ці дані через MQTT-брокер. Додаток отримує виміри, оцінює позицію кожного вузла у просторі приміщення і формує теплову карту покриття.

Вузли не потребують дротового з'єднання між собою. Після первинного налаштування вся комунікація відбувається через існуючу Wi-Fi мережу і MQTT-брокер.

Перед початком роботи кожен підлеглий вузол повинен отримати облікові дані Wi-Fi мережі і MQTT-брокера. Робити це вручну на кожному пристрої незручно, тому я реалізував автоматичну передачу конфігурації через BLE.

Головний вузол після запуску підіймає власну Wi-Fi точку доступу. Користувач підключається до неї з будь-якого пристрою, відкриває браузер і вводить назву і пароль локальної мережі та облікові дані MQTT-брокера. Після цього головний вузол починає транслювати BLE-рекламу.

Кожен підлеглий вузол при запуску сканує ефір у пошуку майстра. Знайшовши його, вузол підключається через BLE і отримує конфігурацію у зашифрованому вигляді. Перед передачею сторони виконують обмін ключами на основі криптографії еліптичних кривих, після чого дані шифруються симетричним алгоритмом AES. Пароль у відкритому вигляді по ефіру не передається жодного разу.

Зми	Арк	№ докум	Підпис	Дата

ВКРБ-123.26.0060.00.00.ПЗ

Арк.

19

Після успішного отримання конфігурації підлеглий вузол повідомляє майстра про завершення процесу і підключається до Wi-Fi та MQTT самостійно. Цей процес займає від 10 до 30 секунд залежно від якості BLE-з'єднання.

Після налаштування всі три вузли працюють незалежно і паралельно. Кожен раз на секунду вузол виконує два вимірювання. Перше це рівень сигналу Wi-Fi від точки доступу, до якої вузол підключений. Друге це рівень BLE-сигналу від двох інших вузлів, який вузол отримує шляхом короткого сканування ефіру.

Результати кожного вузла публікуються в один і той самий MQTT-топик у форматі JSON. Головний вузол повідомляє свій Wi-Fi RSSI і BLE RSSI двох підлеглих. Кожен підлеглий вузол повідомляє свій Wi-Fi RSSI і BLE RSSI головного та однорангового підлеглого.

Таким чином, за один повний цикл система збирає дев'ять значень RSSI. Три Wi-Fi виміри, по одному від кожного вузла до точки доступу, і шість BLE вимірів між парами вузлів. Оскільки кожна пара вузлів вимірює сигнал в обох напрямках, кожна з трьох міжвузлових відстаней підтверджена двома незалежними вимірюваннями.

Один повний цикл з урахуванням часу BLE-сканування займає приблизно 3-4 секунди на кожному вузлі. Оскільки вузли не синхронізовані між собою, додаток орієнтується не на час, а на факт отримання даних від усіх трьох. Як тільки надійшли пакети від головного і обох підлеглих, запускається обробка.

RSSI від ESP32 нестабільний. Навіть у нерухомому приміщенні без змін у середовищі виміри одного і того ж сигналу можуть відрізнятися на 5-10 dBm між сусідніми пакетами. Якщо передавати ці сирі значення безпосередньо в алгоритм оцінки позицій, розраховані координати вузлів будуть помітно стрибати на карті навіть при нерухомих вузлах.

Змн	Арк	№ докум	Підпис	Дата

Тому додаток застосовує до кожного вхідного значення експоненційне згладжування. Кожне нове вимірювання не замінює попереднє повністю, а змішується з ним у заданій пропорції. Це гасить короточасні флуктуації і дає стабільніший сигнал для подальших розрахунків.

При цьому система розрізняє флуктуацію і реальну зміну. Якщо нове значення різко відрізняється від попереднього, наприклад через фізичне переміщення вузла, нове значення приймається з більшою вагою, щоб система швидко реагувала на зміну умов. Якщо значення відсутнє через те, що BLE-сканування не виявило вузла в конкретному циклі, замість нуля використовується попереднє згладжене значення.

Оцінка позицій вузлів є основною обчислювальною задачею системи. На вхід подаються дев'ять згладжених RSSI і координата точки доступу на сітці, яку ви позначаєте клацанням миші в додатку. На виході три точки на сітці, які позначають приблизне розташування кожного вузла.

Перетворення RSSI у відстань виконується за моделлю логарифмічних втрат на шляху поширення сигналу. Ця модель пов'язує рівень сигналу з відстанню через два параметри: опорний рівень сигналу на відстані 1 метр і показник загасання, який залежить від властивостей середовища. Користувач може змінювати обидва параметри через повзунки у додатку і одразу бачити, як змінюється розташування вузлів на карті.

Wi-Fi відстані від кожного вузла до точки доступу і BLE відстані між парами вузлів разом утворюють систему геометричних обмежень. Задача формулюється як пошук трьох точок у просторі, які найкраще задовольняють ці обмеження одночасно. Оскільки аналітичного рішення для такої системи немає через нелінійність і шум у вимірюваннях, задача вирішується чисельною оптимізацією.

Алгоритм виконує кілька незалежних спроб знайти рішення, кожного разу починаючи з різної початкової точки. Деякі спроби стартують поблизу

результату попереднього циклу, інші досліджують нові ділянки простору. З усіх знайдених рішень обирається те, яке дає найменшу сумарну похибку.

Крім геометричних відстаней, алгоритм враховує додаткові обмеження. Вузли не можуть виявитися в одній точці. Порядок вузлів за відстанню від точки доступу на карті має відповідати порядку за Wi-Fi RSSI, тому що сильніший сигнал означає меншу відстань. Користувач може задати очікуваний напрямок зміщення вузлів від точки доступу через кнопки зміщення в інтерфейсі, якщо він знає приблизне розташування вузлів.

Щоб карта не стрибала між циклами, поточний результат притягується до попереднього з невеликою вагою. При нерухомих вузлах це дає стабільну карту. При переміщенні вузлів новий результат все одно знаходить правильне положення, але плавніше.

Після того як позиції вузлів оцінені, додаток прив'язує Wi-Fi RSSI кожного вузла до його розрахованої координати на сітці. Ці пари значень накопичуються. Кожна нова оцінка не стирає попередні, а додається до них. Для кожної клітинки сітки зберігається сума всіх RSSI і кількість потраплянь. При відображенні використовується середнє значення.

Це означає, що з часом і з більшою кількістю положень вузлів карта стає точнішою і детальнішою. Якщо користувач переміщує вузли по приміщенню крок за кроком і чекає кілька циклів у кожному положенні, карта охоплює більше точок і інтерполяція між ними дає кращий результат.

Клітинки без вимірювань поблизу позицій вузлів заповнюються інтерпольованим значенням через зважений розподіл на основі відстані від сусідніх клітинок з даними. Клітинки зовсім далеко від будь-якого вузла залишаються порожніми. Користувач може накласти на сітку растрове зображення плану приміщення і підібрати масштаб так, щоб стіни і кімнати збіглися з сіткою, і тоді карта покриття стає прив'язаною до реального простору.

Кольорова шкала карти йде від синього при слабкому сигналі до червоного при сильному. Порогові значення шкали фіксовані: мінус 100 dBm як нижня межа і мінус 30 dBm як верхня.

Додаток при запуску шукає файл конфігурації поряд із виконуваним файлом. Якщо файлу немає, він створюється з вбудованими значеннями за замовчуванням. Користувач може відредагувати цей файл і вказати адресу брокера, порт, топик і облікові дані, після чого перезапустити додаток без перекомпіляції. Це зручно при розгортанні системи в середовищах з нестандартними налаштуваннями брокера.

Додаток автоматично веде журнал вимірювань у форматі CSV з початку кожної сесії. Файл зберігається у системній директорії даних застосунку у підпапці logs і отримує ім'я з UTC-міткою часу запуску. Кожен запис містить момент вимірювання, усі дев'ять згладжених RSSI-значень від трьох вузлів, розраховані координати кожного вузла на сітці і відстані між усіма трьома парами вузлів у метрах.

Теплову карту можна зберегти вручну у JSON-файл. Він містить усі накопичені дані клітинок, координати точки доступу і зображення плану приміщення. При завантаженні цього файлу стан карти повністю відновлюється і нові вимірювання додаються до існуючих. Це дозволяє розбивати обстеження на кілька сесій і об'єднувати результати.

3.2 Розробка структурної схеми

Структурна схема системи наведена на рисунку 3.1. Вона показує склад системи, її основні компоненти та зв'язки між ними без деталізації реалізації.

Система складається з п'яти компонентів. Мікроконтролер ESP32 Мастер, два мікроконтролери ESP32 Підлеглі (тут зібрані в один компонент), точка доступу Wi-Fi, MQTT-брокер і десктопна програма. Мастер і підлеглі з'єднані між собою по BLE. Усі три мікроконтролери підключені до точки

Зми	Арк	№ докум	Підпис	Дата

доступу Wi-Fi, яка з'єднана з MQTT-брокером. Десктопна програма підключається до того самого брокера.

Мастер відповідає за два незалежних завдання. Перше, прийом облікових даних від користувача та їх захищена передача підлеглим вузлам. Друге, збір даних про силу сигналу та їх публікація через MQTT.

Підлеглі вузли отримують облікові дані від майстра по BLE-каналі і після цього самостійно підключаються до мережі. Їхня основна роль у системі зводиться до збору вимірювань і публікації даних, аналогічно до майстра.

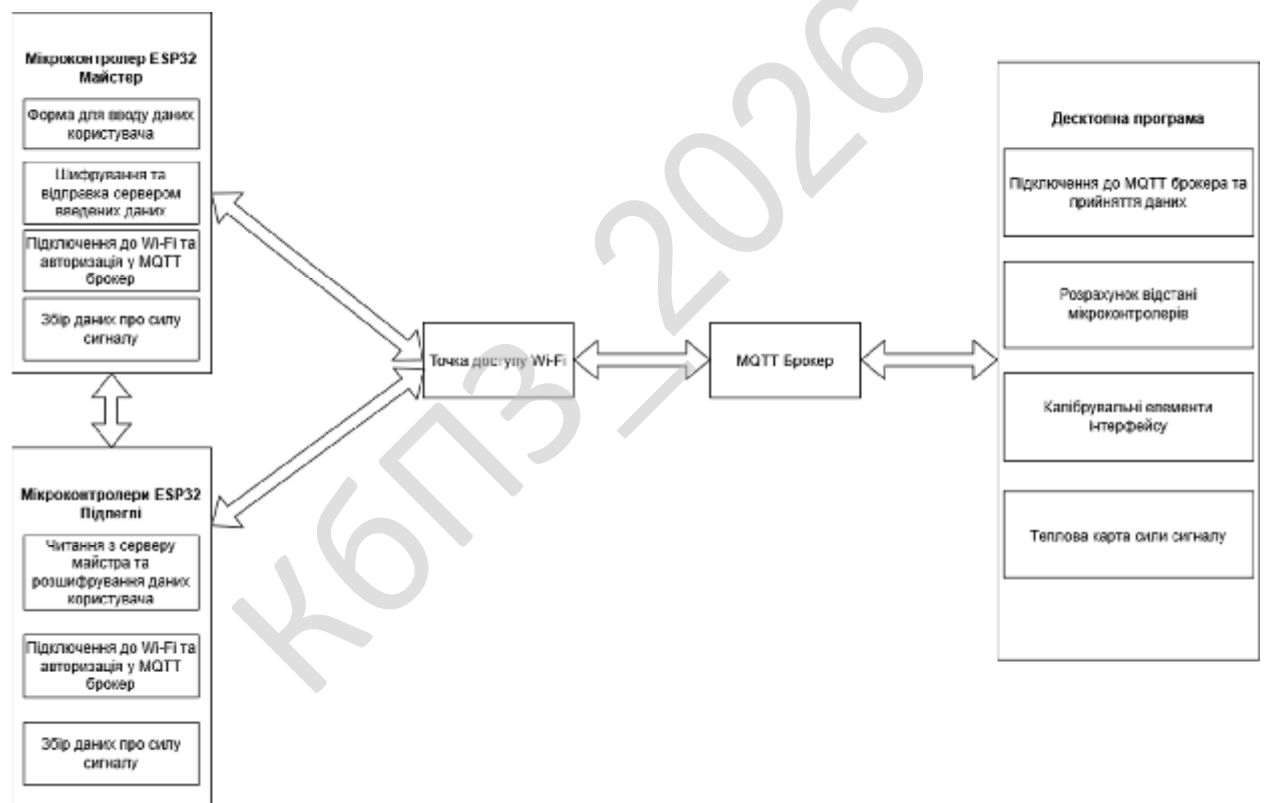


Рисунок 3.1 – Структурна схема системи

Точка доступу Wi-Fi є пасивним елементом інфраструктури. Її сигнал слугує опорною точкою для вимірювань RSSI на стороні мікроконтролерів, а мережевий трафік вона передає до MQTT-брокера без жодної обробки. Брокер робить те саме відносно десктопної програми.

Десктопна програма є кінцевим споживачем даних у системі. Вона приймає вимірювання від усіх трьох вузлів, обчислює їхні позиції відносно точки доступу і відображає результати у вигляді теплової карти. Інтерфейс містить калібрувальні елементи, які дозволяють вам підлаштувати модель розрахунку під конкретне приміщення.

Така архітектура дає одну практичну перевагу: підлеглі вузли не потребують жодного попереднього налаштування з вашого боку. Користувач взаємодіє тільки з майстром, а підлеглі отримують все необхідне від нього по BLE.

3.3 Розробка функціональної схеми

Функціональна схема системи наведена на рисунку 3.2. Вона деталізує поведінку кожного компонента, показує послідовність операцій і напрямки передачі даних між вузлами.

Першою виконується ініціалізація мастера. Він піднімає власну точку доступу Wi-Fi і запускає HTTP-сервер на адресі 192.168.4.1:8080, через який користувач вводить SSID мережі, пароль і параметри MQTT-брокера. Паралельно мастер генерує пару ключів ECDH на кривій secp256r1 і шифрує введені облікові дані за допомогою AES-128-CBC. Зашифровані дані завантажуються на BLE GATT сервер, де кожне поле доступне за окремим UUID. Одночасно мастер починає транслювати власний BLE-ефір, щоб підлеглі вузли могли вимірювати рівень його сигналу.

Підлеглі вузли запускаються в режимі сканування BLE. Кожен з них підключається до GATT сервера майстра, зчитує зашифровані облікові дані за UUID-характеристиками і розшифровує їх локально, використовуючи той самий механізм ECDH з AES-128-CBC. Після отримання даних підлеглий підключається до вказаної Wi-Fi мережі та авторизується в MQTT-брокері. Він

Зми	Арк	№ докум	Підпис	Дата

також піднімає власний BLE-ефір, щоб інші вузли могли вимірювати рівень його сигналу.

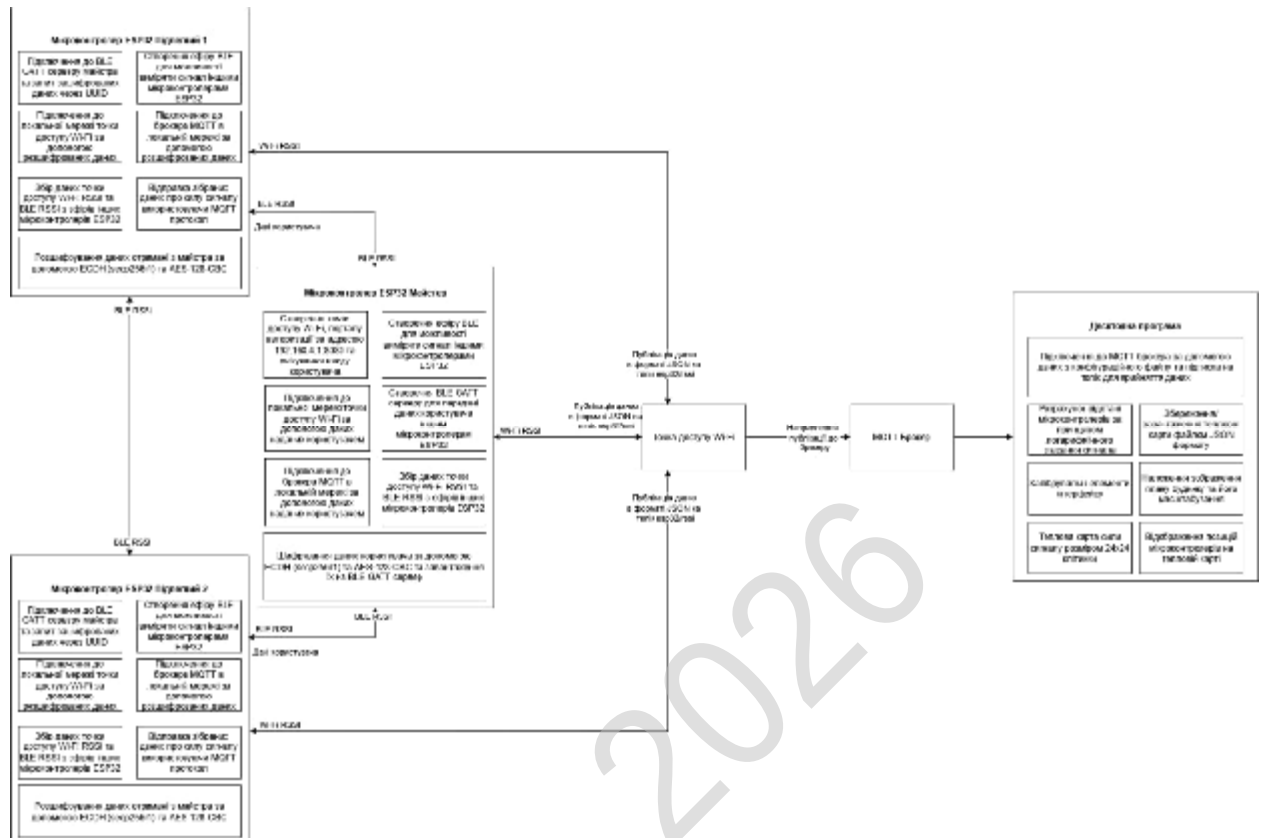


Рисунок 3.2 – Функціональна схема системи

Після того як усі три мікроконтролери підключились до мережі, кожен з них починає циклічно збирати два типи вимірювань. Перший, це рівень Wi-Fi RSSI від точки доступу, що відображає відстань до роутера. Другий, це рівень BLE RSSI від ефірів двох інших мікроконтролерів, що дозволяє оцінити відстань між вузлами. Зібрані значення упаковуються в JSON і публікуються в топик esp32/rssi через MQTT.

Точка доступу Wi-Fi пересилає ці публікації до MQTT-брокера. Брокер, у свою чергу, доставляє їх десктопній програмі, яка підписана на той самий топик.

Десктопна програма отримує JSON-пакети від усіх трьох вузлів і розрізняє їх за полем chip. На основі значень Wi-Fi RSSI вона обчислює

відстань від кожного мікроконтролера до точки доступу за логарифмічною моделлю загасання сигналу. BLE RSSI між вузлами використовується для оцінки відстаней між самими мікроконтролерами. Результати розрахунків відображаються на тепловій карті розміром 24 на 24 клітинки, де позначені розраховані позиції всіх трьох вузлів. Якщо користувач завантажує план будинку, програма накладає його на сітку з можливістю масштабування. Теплова карта зберігається і завантажується у форматі JSON, що дозволяє продовжити сесію вимірювань пізніше без втрати накопичених даних. Калібрувальні елементи інтерфейсу дозволяють вам вручну коригувати опорний рівень RSSI та показник загасання шляху для підвищення точності позиціювання в конкретному приміщенні.

3.4 Розробка діаграми процесів

На рисунку 3.3 зображена розроблена діаграма процесів. Я побудував цю діаграму процесів так, щоб вона показувала роботу системи як послідовність станів скінченного автомата. Кожна вершина відповідає окремій дії, а перехід до наступного стану відбувається після завершення поточної операції.

Робота системи починається з процесу «Початок». Далі виконується «Ініціалізація системи». На цьому етапі система налаштовує апаратні компоненти ESP32-вузлів, ініціалізує криптографічні примітиви ECDH, AES і SHA-256, генерує пари ключів для кожного вузла та запускає Wi-Fi точку доступу на мастер-вузлі для подальшого налаштування.

Після завершення ініціалізації система переходить до процесу «Конфігурація системи». Через HTTP-портал вводяться параметри Wi-Fi мережі та брокера MQTT. Коли конфігурація отримана, запускається процес «Обмін ключами». У ньому вузлові пристрої сканують BLE-рекламу мастер-пристрою, зчитують його відкритий ключ, надсилають власний відкритий

Зми	Арк	№ докум	Підпис	Дата

ключ і на основі спільного секрету ECDH отримують зашифровані дані конфігурації. Після розшифрування і підтвердження АСК цей етап завершується.

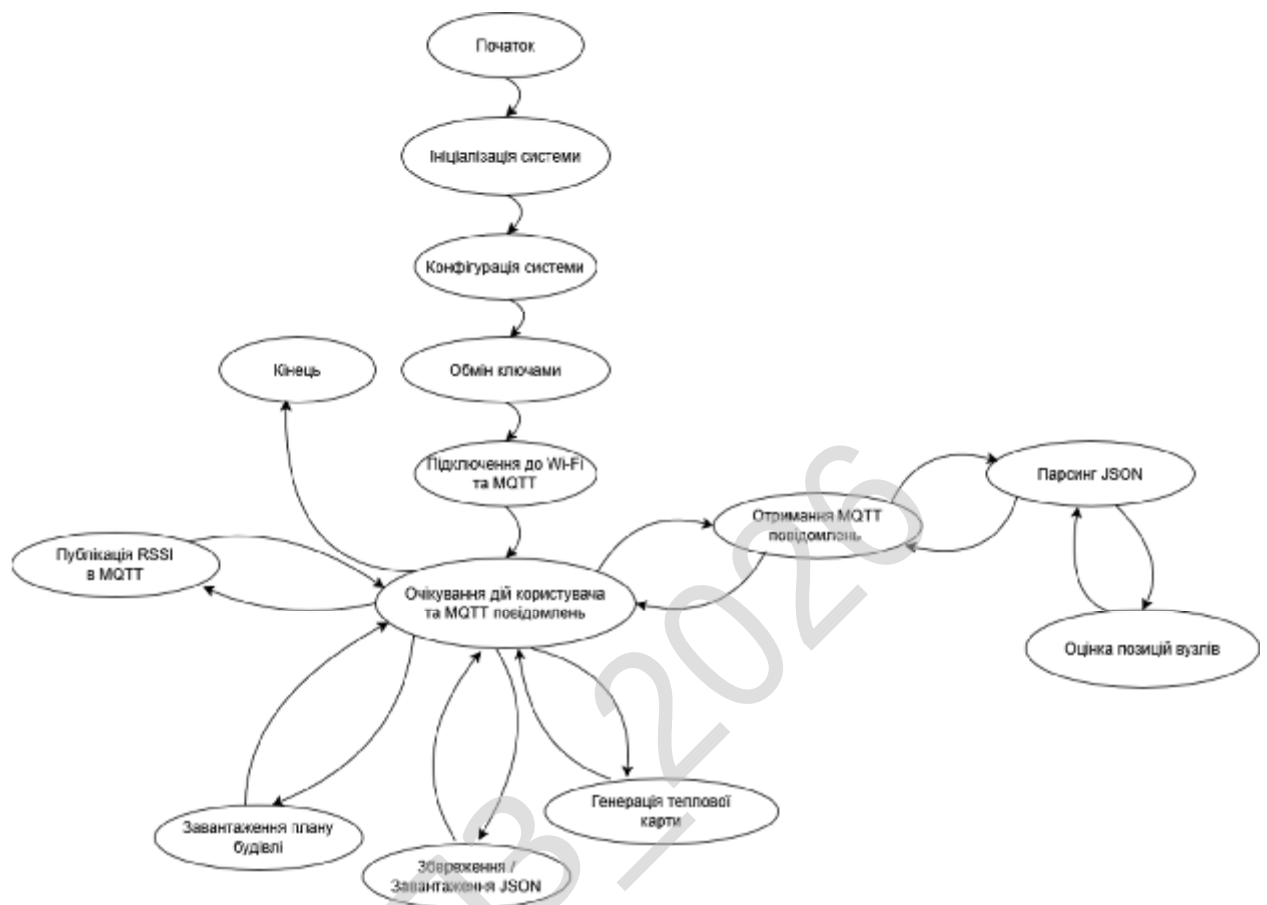


Рисунок 3.3 Діаграма взаємодії процесів

Далі система переходить до процесу «Підключення до Wi-Fi та MQTT». Тут кожен вузол підключається до бездротової мережі та встановлює з'єднання з брокером MQTT. Після цього керування переходить до центрального циклічного процесу «Очікування дій користувача та MQTT повідомлень». Саме він працює як вузол керування всією системою і приймає переходи від підпорядкованих процесів.

Процес «Публікація RSSI в MQTT» відповідає за циклічне зчитування значень RSSI з Wi-Fi і BLE на кожному ESP32-вузлі та публікацію цих даних

у форматі JSON у відповідний топик MQTT. Після цього система повертається до центрального процесу.

Процес «Отримання MQTT повідомлень» використовується десктопним застосунком, який підписується на відповідний топик і приймає повідомлення від усіх вузлів. Після надходження чергового пакета виконується перехід до процесу «Парсинг JSON». Тут додаток розбирає вхідні дані, витягує поля chip, wifi, ble1, ble2 або ble_master, ble_peer та перевіряє, чи коректна структура повідомлення. Далі запускається процес «Оцінка позицій вузлів». У ньому за допомогою зваженої трилатерації та стохастичного оптимізатора обчислюються координати мастер-вузла і двох підлеглих пристроїв на сітці. Після обчислень результат повертається до центрального процесу для відображення на тепловій карті.

Процес «Генерація теплової карти» формує карту покриття сигналу на основі накопичених даних про позиції пристроїв і значень RSSI. Для цього використовується логарифмічне згасання сигналу від точки доступу та BLE. Після оновлення зображення керування повертається до центрального процесу.

Процес «Завантаження плану будівлі» дозволяє користувачу в будь-який момент завантажити растровий план приміщення і накласти його поверх теплової карти. Після завантаження та масштабування зображення процес завершується, а керування повертається до центрального стану.

Процес «Збереження / Завантаження JSON» дає змогу зберегти або відновити стан теплової карти, положення точки доступу та зображення плану будівлі у форматі JSON. Після завершення операції система також повертається до центрального процесу.

Отже, діаграма процесів показує дві частини роботи системи. Перша частина охоплює послідовну ініціалізацію від «Початку» до «Очікування». Друга частина є циклічною і описує, як центральний процес координує отримання даних, обчислення позицій і дії користувача.

Зми	Анк	№ докум	Підпис	Дата

4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ

4.1 Розробка блок-схем та опис алгоритмів функціонування системи

Система має два рівні алгоритмів. Перший рівень це мікропрограми ESP32, які відповідають за збір сигналів і їх передачу. Другий рівень це Qt-додаток, який отримує ці дані, фільтрує їх, обчислює позиції вузлів і формує теплову карту. Ці два рівні взаємодіють виключно через MQTT-брокер, і це накладає певні обмеження на алгоритм: Qt-додаток не може запитати в ESP32 «надішли ще раз» або «ти ще живий». Він просто отримує або не отримує пакети.

Блок-схема основної програми наведена на рисунку 4.1. Вона відображає повний життєвий цикл Qt-додатку від запуску до завершення.

Ініціалізація починається з пошуку файлу mqtt_config.ini. Пошук виконується спочатку в директорії виконуваного файлу, потім у поточній робочій директорії. Якщо файл знайдений, зчитуються всі параметри підключення. Якщо немає, файл створюється з вбудованими значеннями за замовчуванням, і далі використовуються саме ці значення.

```
const QString appConfig = QApplication::applicationDirPath() +
QStringLiteral("/mqtt_config.ini");
const QString cwdConfig = QDir::currentPath() +
QStringLiteral("/mqtt_config.ini");

QFileInfo fi(appConfig);
if (fi.exists() && fi.isFile()) {
    cfg.sourcePath = appConfig;
} else {
    fi.setFile(cwdConfig);
```

```

if (fi.exists() && fi.isFile()) {
    cfg.sourcePath = cwdConfig;
} else {
    cfg.sourcePath = appConfig;

```

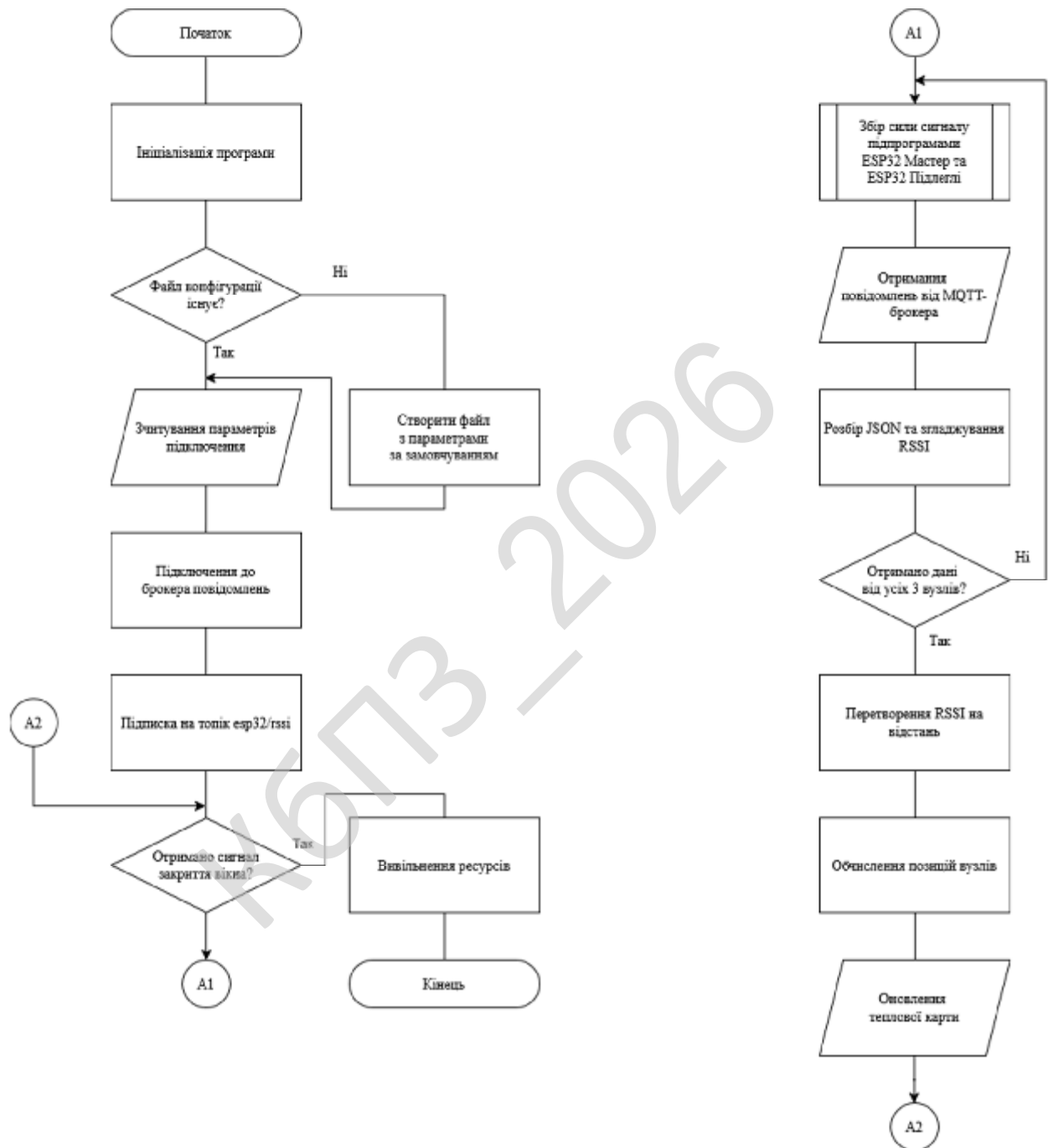


Рисунок 4.1 Блок-схема основної програми

```

writeDefaultConfigIfMissing(cfg.sourcePath, defaultHost, defaultPort,
defaultTopic);
return cfg;

```

Змн	Арх	№ докум	Підпис	Дата

```
}  
}
```

Після зчитування конфігурації додаток підключається до MQTT-брокера і підписується на топик esp32/rssi. Також з цим ініціалізується лог-файл: він створюється в підпапці logs поряд з виконуваним файлом, отримує ім'я з UTC-міткою часу поточної сесії і відразу записується заголовок CSV. Якщо папка logs не існує, вона створюється автоматично через QDir::mkpath.

Після ініціалізації додаток переходить у режим очікування подій. Тут можливо два варіанти. Або надходить MQTT-повідомлення від одного з вузлів, або користувач щось робить в інтерфейсі. При надходженні сигналу закриття вікна виконується завершення: лог-файл скидається на диск через flush() і закривається, після чого знищуються об'єкти інтерфейсу.

Обробка повідомлень і накопичення циклу

Кожне вхідне MQTT-повідомлення передається в метод onMqttMessageReceived. Спочатку перевіряється ім'я топика, потім повідомлення передається у ChipDataModel::updateFromPayload. Цей метод розбирає JSON, визначає, від якого вузла прийшло повідомлення, і оновлює згладжені значення RSSI для цього вузла. Якщо JSON некоректний або відсутні обов'язкові поля, метод повертає false і в рядку стану виводиться текст помилки з першими 100 символами проблемного пакету.

Якщо розбір успішний, викликається consumeUpdateAndCheckCycleReady. Метод додає ім'я поточного вузла до множини m_cycleSeen і перевіряє, чи є там MASTER і хоча б два вузли з іншими іменами. Якщо умова виконана, множина очищається і метод повертає true, що означає готовність до повного циклу обробки.

Я вибрав саме такий підхід, а не обробку по таймеру, так як ESP32-вузли не синхронізовані між собою, і різниця в часі між їхніми пакетами може становити від кількох сотень мілісекунд до 2-3 секунд залежно від часу BLE-сканування. Якщо запускати обробку по таймеру, часто буде ситуація, коли

Зми	Арк	№ докум	Підпис	Дата

один із вузлів ще не встиг надіслати черговий пакет, і алгоритм використовуватиме дані з різних моментів часу. На практиці це давало помітні стрибки маркерів при нерухомих вузлах. Очікування пакету від кожного вузла вирішило це.

Блок-схеми підпрограм ESP32 наведені на рисунку 4.2. Вони показують чотири окремих алгоритми: ініціалізацію та основний цикл для мастера і те саме для підлеглого вузла.

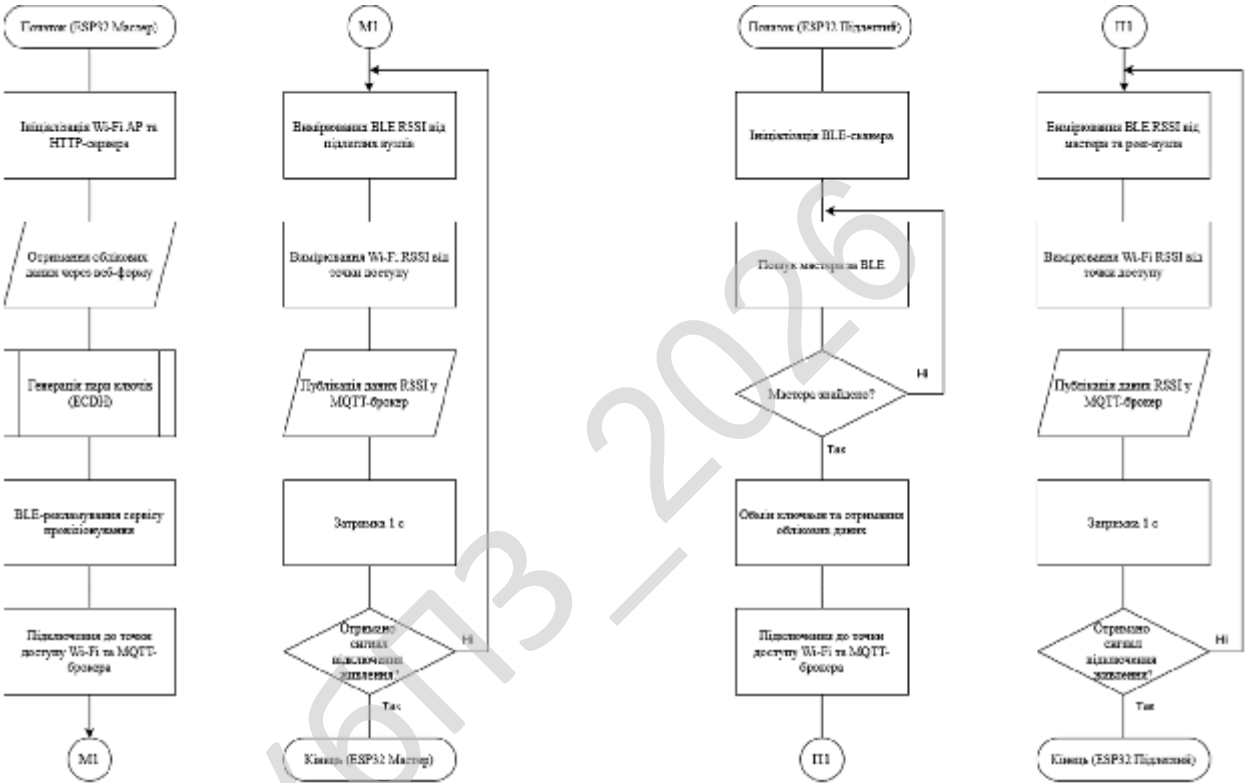


Рисунок 4.2 Блок-схема підпрограм ESP32

Алгоритм мастера починається з одночасного виконання двох завдань. Перше це підготовка до провізйонування підлеглих. Друге це власне підключення до мережі. На початку мастер піднімає власну точку доступу і HTTP-сервер, через який приймає облікові дані мережі і брокера. Одночасно він генерує пару ECDH-ключів. Після отримання даних через веб-форму мастер запускає BLE-рекламування і підключається до Wi-Fi та MQTT сам.

Підлеглий вузол стартує в режимі активного сканування BLE. Він шукає пристрій, що рекламує відомий UUID сервісу провізіїонування. До знаходження майстра сканування виконується у циклі з затримкою 250 мс між спробами. Щойно пристрій знайдено, підлеглий підключається до нього по BLE і виконує процедуру обміну ключами.

Алгоритм провізіїонування мікроконтролерів

Процес передачі конфігурації від майстра до підлеглих вузлів реалізований через BLE з криптографічним захистом каналу. Після отримання облікових даних через HTTP-форму майстер генерує пару ключів і починає рекламувати BLE-сервіс провізіїонування.

Підлеглий знаходить майстра за UUID сервісу, підключається до нього і виконує обмін ключами. Після цього майстер передає зашифрований рядок конфігурації формату CFGv1|ssid|password|mqttHost|mqttUser|mqttPass|END через BLE-характеристику. Підлеглий розшифровує його, перевіряє, чи всі поля присутні і непорожні, і надсилає байт підтвердження 0x01 у відповідь. Якщо перевірка не пройшла, надсилається 0x00 і підлеглий повертається до стану пошуку. Криптографічна частина цього процесу детальніше описана в підрозділі 4.2.

Після підтвердження підлеглий підключається до Wi-Fi і MQTT, після чого переходить в основний робочий цикл: сканування BLE, вимірювання Wi-Fi RSSI, публікація пакету кожену секунду.

Одна практична складність, яку я зустрів під час розробки: BLE-з'єднання між двома ESP32 іноді розривалось на етапі запису ключа підлеглим. Це траплялось приблизно в одному випадку з десяти. Вирішив додавши затримку delay(500) між записом ключа і читанням зашифрованого рядка. Цього виявилось достатньо, щоб майстер встигав обробити запис і оновити характеристику до того, як підлеглий намагається її зчитати.

Зми	Анк	№ докум	Підпис	Дата

Згладжування вхідних даних

Після розбору JSON кожне значення RSSI проходить через фільтр експоненційного згладжування. Для кожного вузла і кожного каналу зв'язку окремо підтримується поточне згладжене значення. При надходженні нового виміру воно оновлюється за формулою (4.1):

$$RSSI_s(t) = RSSI_s(t - 1) + \alpha * (RSSI(t) - RSSI_s(t - 1)) \quad (4.1)$$

де $RSSI_s(t)$ це згладжене значення на поточному кроці,

$RSSI_s(t-1)$ це значення на попередньому кроці,

$RSSI(t)$ це новий сирий вимір,

α це коефіцієнт згладжування від 0 до 1. При α рівному 1 фільтр відключається повністю. При α близькому до 0 фільтр майже не реагує на нові виміри.

Значення α за замовчуванням я встановив рівним 0,6. Це результат спостережень під час тестування: при $\alpha = 0,3$ карта була стабільна, але при переміщенні вузлів позначка тяглась за вузлом із запізненням у 5-6 циклів. При $\alpha = 0,8$ реакція була швидка, але від шуму 5-7 дБм координати стрибали на 1-2 клітинки навіть при нерухомих вузлах. Значення 0,6 дало прийнятний компроміс.

Є важливий виняток для різких стрибків сигналу. Якщо різниця між новим виміром і поточним згладженим значенням перевищує 15 дБм хоча б по одному каналу, коефіцієнт примусово встановлюється в 1,0 тільки для цього оновлення.

```
const double jumpWifi = zeroWifi ? 0.0 : qAbs(mergedWifi - prev.at(0));
const double jumpA    = zeroA    ? 0.0 : qAbs(mergedA    - prev.at(1));
const double jumpB    = zeroB    ? 0.0 : qAbs(mergedB    - prev.at(2));
const double alpha = (jumpWifi > 15.0 || jumpA > 15.0 || jumpB > 15.0) ? 1.0
: m_rssiSmoothingAlpha;
nextValues = QVector<double>{
    zeroWifi ? prev.at(0) : prev.at(0) + alpha * (mergedWifi - prev.at(0)),
```

```

zeroA    ? prev.at(1) : prev.at(1) + alpha * (mergedA    - prev.at(1)),
zeroB    ? prev.at(2) : prev.at(2) + alpha * (mergedB    - prev.at(2))
};

```

Стрибок на 15 дБм за один цикл у нерухомому середовищі малоймовірний. Якщо таке відбулось, це або фізичне переміщення вузла, або поява перешкоди. В обох випадках реакція має бути миттєвою, а не розтягнутою на кілька циклів.

Окремо обробляються нульові значення. ESP32 передає 0 для BLE-каналу, якщо в поточному скануванні не виявив сусідній вузол. Нуль не є виміром RSSI, тому він замінюється на попереднє збережене значення. Якщо попереднього значення ще немає, використовується мінус 100 дБм як значення за замовчуванням, що фактично означає «вузол поза зоною видимості» і виключає його з розрахунків позиції.

Перетворення сигналу у відстань

Після отримання згладжених RSSI для кожного каналу обчислюється відстань у метрах за моделлю логарифмічного загасання сигналу (4.2):

$$d = 10^{\frac{(RSSI_0 - RSSI)}{(10n)}} \quad (4.2)$$

де d це розрахована відстань у метрах,

$RSSI_0$ це опорний рівень сигналу на відстані 1 метр від джерела,

$RSSI$ це поточне згладжене значення,

n це показник загасання, що залежить від властивостей середовища.

В реалізації є два незалежних набори параметрів. Для Wi-Fi $RSSI_0$ за замовчуванням мінус 41,8 дБм, для BLE мінус 68,7 дБм. Показник загасання n однаковий для обох і за замовчуванням дорівнює 2,5. Я підібрав ці значення експериментально: розміщував вузли на відомих відстанях від роутера і підбирав параметри так, щоб розрахована відстань збігалась з реальною. Для конкретного приміщення з іншою плануванням або іншим обладнанням

значення можуть відрізнитись, тому всі три параметри виведені в інтерфейс як повзунки.

```
double MainWindow::rssiToDistance(double rssi) const
{
    return qPow(10.0, (m_referenceRssi - rssi) / (10.0 *
m_pathLossExponent));
}
double MainWindow::bleRssiToDistance(double rssi) const
{
    return qPow(10.0, (m_bleReferenceRssi - rssi) / (10.0 *
m_pathLossExponent));
}
```

BLE відстань між парою вузлів обчислюється як середнє двох симетричних вимірювань. Вузол А вимірює сигнал від В, вузол В вимірює сигнал від А. Результат усереднюється перед підстановкою у формулу. Це частково компенсує несиметричність антен: якщо антена одного вузла спрямована в протилежний бік і дає занижений вимір, антена другого вузла цей вимір частково виправляє.

```
const double rssi01 = (sanitizeRssiSign(values.at(1)) +
sanitizeRssiSign(values.at(4))) / 2.0;
const double rssi02 = (sanitizeRssiSign(values.at(2)) +
sanitizeRssiSign(values.at(7))) / 2.0;
const double rssi12 = (sanitizeRssiSign(values.at(5)) +
sanitizeRssiSign(values.at(8))) / 2.0;
```

На виході цього кроку є шість відстаней у метрах, які потім діляться на `m_metersPerCell` і переводяться у клітинки сітки. Параметр `metersPerCell` задається користувачем через повзунок в інтерфейсі. Це відстань у реальному просторі, яку представляє одна клітинка сітки 24x24. При значенні 0,5 метра вся сітка охоплює приміщення 12x12 метрів, при 1 метрі охоплення 24x24 метри.

При зміні `metersPerCell` через повзунок повністю очищується накопичена теплова карта і скидаються всі кеші позицій. Це потрібно тому, що всі раніше накопичені координати клітинок відповідали попередньому масштабу і після зміни стають некоректними.

Алгоритм оцінки позицій

Маючи шість відстаней у клітинках і координату точки доступу на сітці, алгоритм `estimateNodePositions` шукає координати трьох вузлів. Задача не має аналітичного рішення через шум у вимірюваннях і кількість невідомих, тому вона зводиться до мінімізації функції помилки.

Функція `computePositionError` приймає три точки і параметри задачі, повертає скалярну оцінку якості рішення. Менше значення означає краще рішення. Функція складається з кількох незалежних блоків штрафів.

Перший блок рахує відхилення між геометричною відстанню від кандидатної позиції вузла до точки доступу і відстанню, яку дало перетворення Wi-Fi RSSI за формулою (4.2). Якщо кандидатна позиція правильна, ці два числа збігаються і блок дає нуль. Якщо вузол стоїть не там, різниця дає ненульовий штраф.

```
error += m_apWeightMaster * qPow(cellDist(points[0], params.apCell) -  
params.dAp0, 2.0);  
error += m_apWeightNodes * qPow(cellDist(points[1], params.apCell) -  
params.dAp1, 2.0);  
error += m_apWeightNodes * qPow(cellDist(points[2], params.apCell) -  
params.dAp2, 2.0);
```

Другий блок це відхилення від BLE відстаней між вузлами. Ваговий коефіцієнт тут значно більший за Wi-Fi. Я підняв його після спостереження: при малій вазі цього блоку оптимізатор знаходив рішення, де всі три вузли розташовувались на правильних Wi-Fi відстанях від точки доступу, але майже в одній точці або в довільному порядку відносно один одного. Після збільшення ваги алгоритм почав поважати взаємні відстані між вузлами і результат став реалістичнішим.

```
error += m_bleDirectedWeight * qPow(d01 - params.targetD01, 2.0);  
error += m_bleDirectedWeight * qPow(d02 - params.targetD02, 2.0);  
error += m_bleDirectedWeight * qPow(d12 - params.targetD12, 2.0);
```

Третій блок це штраф за порушення мінімальної відстані між вузлами. Якщо два вузли ближче ніж 1 клітинка, штраф зростає зі зменшенням відстані.

Зми	Анк	№ докум	Підпис	Дата

Без цього при сильно зашумлених даних оптимізатор іноді зводив два вузли в одну точку, що є малоімовірним.

```
const double violation = minD - dist;
return violation > 0.0 ? violation * violation : 0.0;
error += kMinNodeSepWeight * minSepPenalty(d01, kMinNodeSepCells);
error += kMinNodeSepWeight * minSepPenalty(d02, kMinNodeSepCells);
error += kMinNodeSepWeight * minSepPenalty(d12, kMinNodeSepCells);
```

Четвертий блок це порядкові умови. Якщо за Wi-Fi RSSI відомо, що вузол А ближчий до точки доступу ніж вузол В, то і на карті А має бути розміщений ближче. Порушення цього правила штрафується найбільшим ваговим коефіцієнтом у всій функції. Порядок вузлів за відстанню від точки доступу визначається достатньо надійно навіть при зашумленому сигналі.

```
error += kOrderWeight * (params.dAp0 < params.dAp1 ? ordErr(act0, act1) :
ordErr(act1, act0));
error += kOrderWeight * (params.dAp0 < params.dAp2 ? ordErr(act0, act2) :
ordErr(act2, act0));
error += kOrderWeight * (params.dAp1 < params.dAp2 ? ordErr(act1, act2) :
ordErr(act2, act1));
```

П'ятий блок штрафує ситуацію, коли майстер опиняється нижче підлеглих вузлів по вертикальній осі сітки. Це пов'язано з обмеженням системи, а саме як відображаються підписи на тепловій карті. Оскільки верх сітки позначений як північ, майстер має бути розміщений північніше підлеглих, інакше маркери М, N1, N2 плутаються між собою.

```
double MainWindow::northViol(double masterY, double nodeY) const
{
    const double violation = masterY - nodeY;
    return violation > 0.0 ? violation * violation : 0.0;
}
```

```
error += kNorthPenaltyWeight * northViol(points[0].y(), points[1].y());
error += kNorthPenaltyWeight * northViol(points[0].y(), points[2].y());
```

Шостий блок це стабілізація відносно попереднього результату. При нерухомих вузлах він дає плавну карту без стрибків між циклами. При

переміщенні вузлів новий результат поступово зміщується до правильного положення.

```
if (m_hasLastEstimate && m_lastEstimatedNodes.size() == 3) {  
    error += m_masterStabilityWeight * qPow(cellDist(points[0],  
        m_lastEstimatedNodes[0]), 2.0);  
    error += kNodeStabilityNudge * qPow(cellDist(points[1],  
        m_lastEstimatedNodes[1]), 2.0);  
    error += kNodeStabilityNudge * qPow(cellDist(points[2],  
        m_lastEstimatedNodes[2]), 2.0);  
}
```

Сьомий блок це зміщення по горизонталі і вертикалі. Користувач може задати очікуваний напрямок розташування вузлів відносно точки доступу через кнопки в інтерфейсі: захід, нейтраль, схід і північ, нейтраль, південь. При виборі зміщення по осі X до кожного вузла додається штраф за відхилення від очікуваної горизонтальної позиції.

```
if (qAbs(m_ewBias) > 0.01) {  
    for (int i = 0; i < 3; ++i) {  
        const double preferredX =  
            params.apCell.x() + m_ewBias * dApArr[i];  
        error += 4.0 * qPow(points[i].x() - preferredX, 2.0);  
    }  
} else {  
    const double centroidX = (points[0].x() + points[1].x() + points[2].x())  
        / 3.0;  
    error += 2.0 * qPow(centroidX - params.apCell.x(), 2.0);  
}
```

Якщо зміщення не задане, діє слабка умова, яка притягує центр мас трьох вузлів до горизонтальної координати точки доступу. Без неї при симетричних вимірюваннях всі три вузли могли зміщуватись далеко вбік від точки доступу і залишатись там, бо функція помилки від цього не зростала. Аналогічна логіка реалізована для вертикальної осі через параметр m_nsBias.

Сам процес мінімізації виглядає так. Запускається 25 рестартів оптимізатора. Кожен рестарт починається з ініціалізації трьох кандидатних точок і виконує 3000 ітерацій покрокового спуску.

```
for (int restart = 0; restart < kOptimizerRestarts; ++restart) {
```

```

QVector<QPointF> candidates(3);
const bool usePrev = m_hasLastEstimate && m_lastEstimatedNodes.size() ==
3 && restart < (kOptimizerRestarts - kOptimizerFreshRestarts);
if (usePrev) {
    for (int i = 0; i < 3; ++i) {
        const double dx =
            (rng->generateDouble() * 2.0 - 1.0) * 3.0;
        const double dy =
            (rng->generateDouble() * 2.0 - 1.0) * 3.0;
        candidates[i] = clampPoint(QPointF(
            m_lastEstimatedNodes[i].x() + dx,
            m_lastEstimatedNodes[i].y() + dy));
    }
} else {
    const double angle = (restart / 25.0) * 2.0 * kPi;
    candidates[0] = clampPoint(QPointF(
        apCell.x() + dAp0 * qCos(angle),
        apCell.y() + dAp0 * qSin(angle)));
    candidates[1] = clampPoint(QPointF(
        apCell.x() + dAp1 * qCos(angle + kSeedAngleStep1),
        apCell.y() + dAp1 * qSin(angle + kSeedAngleStep1)));
    candidates[2] = clampPoint(QPointF(
        apCell.x() + dAp2 * qCos(angle + kSeedAngleStep2),
        apCell.y() + dAp2 * qSin(angle + kSeedAngleStep2)));
}
}

```

Перші 13 рестартів ініціалізуються навколо попереднього результату зі збуренням до 3 клітинок. Останні 12 використовують детерміновану кутову ініціалізацію навколо точки доступу на радіусах, що відповідають Wi-Fi відстаням. Рестарти навколо попереднього результату добре знаходять точне рішення поблизу правильного, але можуть застрягти в локальному мінімумі. Свіжі рестарти з кутовою ініціалізацією дозволяють виявити принципово інше рішення, якщо попереднє було хибним.

На кожній ітерації випадково обирається один із трьох вузлів і до нього застосовується випадкове зміщення з поточним кроком step. Якщо нова позиція дає меншу помилку, вона приймається.

```
double step = kOptimizerInitialStep;
```

```

double currentScore = computePositionError(candidates, params);
for (int iter = 0; iter < kOptimizerIterations; ++iter) {
    const int idx = rng->bounded(3);
    QVector<QPointF> trial = candidates;
    const double dx = (rng->generateDouble() * 2.0 - 1.0) * step;
    const double dy = (rng->generateDouble() * 2.0 - 1.0) * step;
    trial[idx] = clampPoint(QPointF(candidates[idx].x() + dx,
    candidates[idx].y() + dy));
    const double trialScore = computePositionError(trial, params);
    if (trialScore < currentScore) {
        candidates = trial;
        currentScore = trialScore;
    }
    step = qMax(kOptimizerMinStep, step * kOptimizerStepDecay);
}

```

Крок `step` починається з 4,0 клітинки і множиться на 0,9994 при кожній ітерації. За 3000 ітерацій він зменшується приблизно до 0,165 клітинки. Мінімальне значення кроку обмежено знизу до 0,05. Це дозволяє досягати точності краще ніж 0,1 клітинки при стабільному сигналі.

Після всіх 25 рестартів береться рішення з найменшою функцією помилки. Воно зберігається як `m_lastEstimatedNodes` і використовується в наступному циклі як початкова точка для рестартів навколо попереднього результату.

Формування теплової карти

Щойно отримані координати вузлів, кожна з них округлюється до найближчої клітинки сітки і до неї додається поточне Wi-Fi RSSI відповідного вузла.

```

for (int i = 0; i < estimatedNodes.size(); ++i) {
    int col = qRound(estimatedNodes.at(i).x());
    int row = qRound(estimatedNodes.at(i).y());
    col = qBound(0, col, kGridCols - 1);
    row = qBound(0, row, kGridRows - 1);
    const int idx = row * kGridCols + col;
    m_heatSums[idx] += wifiSamples.at(i);
    m_heatCounts[idx] += 1;
}

```

Значення клітинки при відображенні це просте середнє арифметичне всіх накопичених вимірювань для неї. Якщо вузол кілька разів проходив через одне місце, результати усереднюються автоматично. При повільному переміщенні вузлів через приміщення карта поступово заповнюється, і кожна нова точка уточнює загальну картину.

Клітинки з прямими вимірюваннями відображаються кольором, що відповідає середньому RSSI за шкалою від синього при мінус 100 дБм до червоного при мінус 30 дБм. Клітинки без вимірювань, але в радіусі впливу від сусідніх позицій вузлів, заповнюються через гауссівську інтерполяцію з вагами, що спадають зі збільшенням відстані.

```
const double espRadiusCells = kEspInfluenceRadiusMeters / m_metersPerCell;
const double espSigmaCells = espRadiusCells / 2.5;
for (int i = 0; i < m_samplePositions.size(); ++i) {
    const QPointF &p = m_samplePositions.at(i);
    const double dx = target.x() - p.x();
    const double dy = target.y() - p.y();
    const double dist2 = dx * dx + dy * dy;
    if (qSqrt(dist2) > espRadiusCells) continue;
    const double gaussianWeight =
        qExp(-dist2 / (2.0 * espSigmaCells * espSigmaCells));
    weightedSum += gaussianWeight * m_sampleValues.value(i, kMinRssi);
    totalWeight += gaussianWeight;
}
```

Радіус впливу одного вузла становить 1 метр в реальних координатах, але оскільки він перераховується через `m_metersPerCell`, при зміні масштабу сітки радіус у клітинках змінюється автоматично. Клітинки, що знаходяться далеко від будь-якого вузла, залишаються порожніми.

4.2 Захист розробленого програмного забезпечення

Підлеглі вузли отримують облікові дані Wi-Fi мережі і MQTT-брокера від майстра по BLE. BLE-трафік можна перехопити пасивним скануванням, не взаємодіючи з пристроями. Якщо передавати пароль у відкритому вигляді,

будь-хто з телефоном і BLE-сканером у радіусі кількох метрів може його отримати. Для системи, яка розгортається в реальному приміщенні, це неприйнятно.

Криптографічний захист каналу реалізований через бібліотеку `mbedtls`. Це відкритий модуль з реалізацією стандартних криптографічних примітивів, який входить до складу ESP-IDF і доступний на ESP32 без додаткових залежностей. З його функцій у системі використовуються три: обмін ключами ECDH на кривій `secp256r1`, хешування SHA-256 і симетричне шифрування AES-128-CBC.

Обмін ключами відбувається так. Обидва вузли незалежно генерують пару ключів через `mbedtls_ecp_gen_keypair`. Публічні ключі передаються по BLE у відкритому вигляді, приватні ключі ніколи не залишають пристрій. Після обміну кожна сторона через `mbedtls_ecdh_compute_shared` обчислює спільний секрет зі свого приватного ключа і публічного ключа іншої сторони. Математична властивість ECDH гарантує, що обидві операції дають однаковий результат без передачі секрету по каналу. Перехоплення обох публічних ключів не дає зломиснику можливості відновити спільний секрет без розв'язання задачі дискретного логарифму на еліптичній кривій.

Зі спільного секрету через `mbedtls_sha256` виводяться 16-байтний ключ AES і 16-байтний вектор ініціалізації. Після виведення область пам'яті зі спільним секретом явно обнуляється:

```
uint8_t digest[32];
mbedtls_sha256(sharedSecret, 32, digest, 0);
memcpy(aesKey, digest, 16);
memcpy(aesIV, digest + 16, 16);
memset(sharedSecret, 0, sizeof(sharedSecret));
```

Рядок конфігурації шифрується через `mbedtls_aes_crypt_cbc` з отриманим ключем, передається по BLE у hex-кодуванні і розшифровується підлеглим тим самим модулем. Підлеглий перевіряє валідність hex-рядка до спроби розшифрування, а після нього перевіряє формат і повноту полів. Якщо

хоча б одне поле порожнє, надсилається NACK і процедура повторюється.
Таким чином, пароль Wi-Fi і MQTT-брокера жодного разу не передається по
ефіру у відкритому вигляді.

К6ПЗ_2026

Змн	Арк	№ докум	Підпис	Дата

ВКРБ-123.26.0060.00.00.ПЗ

Арк.
45

5 ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

Запуск системи починається з підключення трьох ESP32-вузлів до живлення. Майстер-вузол піднімає точку доступу з назвою ESP32_SETUP, до якої оператор підключається з будь-якого пристрою і через веб-форму вводить облікові дані мережі Wi-Fi та адресу MQTT-брокера. Після відправки форми майстер підключається до мережі, запускає BLE-рекламу і автоматично передає зашифровані налаштування двом вузлам-підлеглим. Весь процес провізйонування займає менше хвилини і більше не потребує повторення, якщо інфраструктура не змінюється.

Десктопний застосунок запускається на будь-якому комп'ютері в тій самій мережі. На початковому екрані, зображеному на рисунку 5.1, видно порожню теплову сітку 24x24, рядок статусу з повідомленням про стан підключення до MQTT-брокера, панель калібрування з трьома повзунками для опорного RSSI Wi-Fi, опорного RSSI BLE та коефіцієнта загасання шляху, а також кнопки для завантаження плану приміщення і збереження карти. До того моменту, поки оператор не розмістить точку доступу на сітці лівою кнопкою миші, система не намагається розраховувати позиції вузлів і відображає підказку про необхідність цієї дії.

Перед початком вимірювань оператор виконує коротке калібрування. Один з вузлів розміщують на відстані рівно одного метра від точки доступу Wi-Fi і зчитують значення RSSI з панелі діагностики. Це значення виставляється повзунком WiFi опорне. Аналогічно два вузли розміщують на відстані одного метра один від одного для калібрування BLE опорного значення. Коефіцієнт загасання за замовчуванням дорівнює 2.5, що відповідає типовому офісному приміщенню. Для кімнат з товстими бетонними стінами значення варто підняти до 3.0-3.5, для відкритих просторів знизити до 2.0.

Зми	Арк	№ докум	Підпис	Дата

ВКРБ-123.26.0060.00.00.ПЗ

Арк.

46

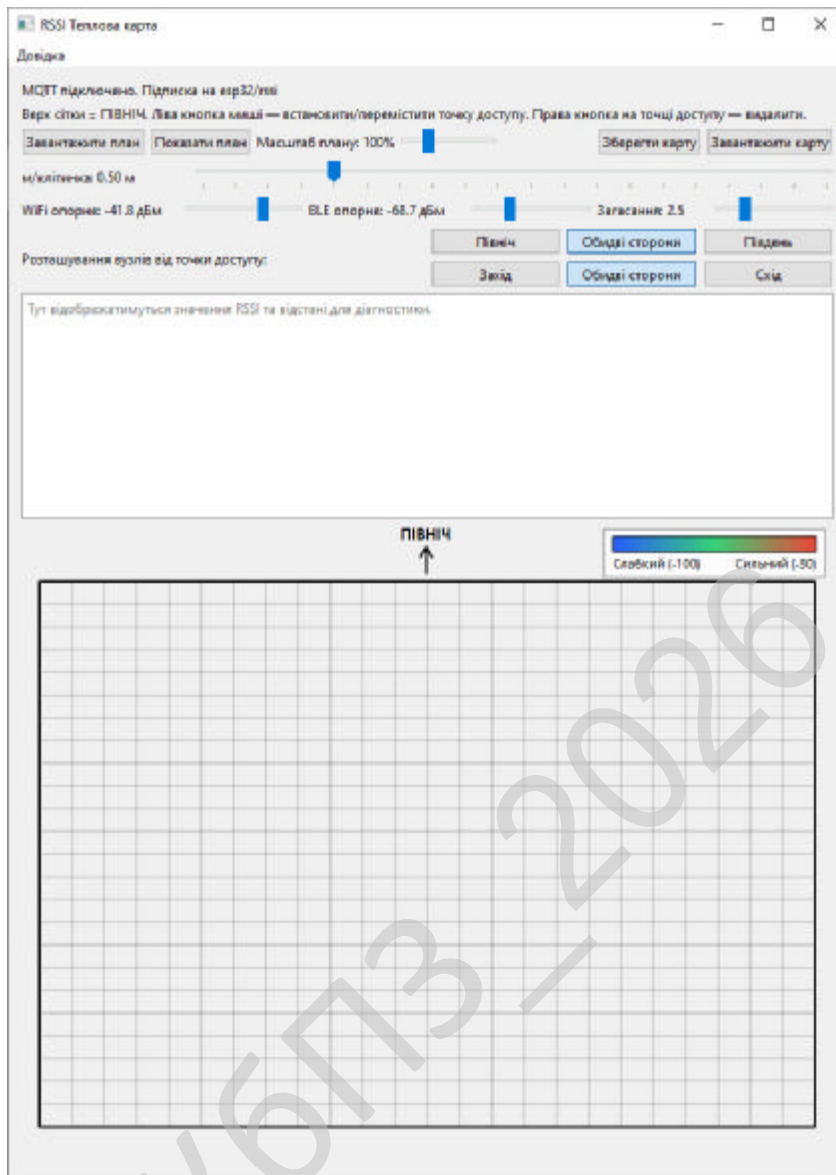


Рисунок 5.1 – Початкове вікно програми

Якщо план приміщення доступний у форматі PNG або JPG, його завантажують кнопкою Завантажити план і вмикають відображення кнопкою Показати план. Повзунок масштабу дозволяє підігнати зображення точно під розміри сітки. У режимі накладення план відображається під тепловою картою з прозорістю, що дозволяє бачити обидва шари одночасно. Це корисно для того, щоб зрозуміти, де стіни або перегородки впливають на сигнал.

Після розміщення точки доступу на сітці система починає приймати RSSI-виміри і поступово заповнює теплову карту. Клітинки, пофарбовані у синій колір, відповідають слабкому сигналу, червоні позначають зони з сильним покриттям. Позначки M, N1 та N2 на сітці показують поточну оцінку розташування трьох вузлів. Накопичені дані можна зберегти у форматі JSON кнопкою Зберегти карту і завантажити пізніше для порівняння з результатами після змін в інфраструктурі.

Меню Довідка містить пункт Авторське право. Відповідне вікно зображене на рисунку 5.2 і містить інформацію про розробника та рік створення.

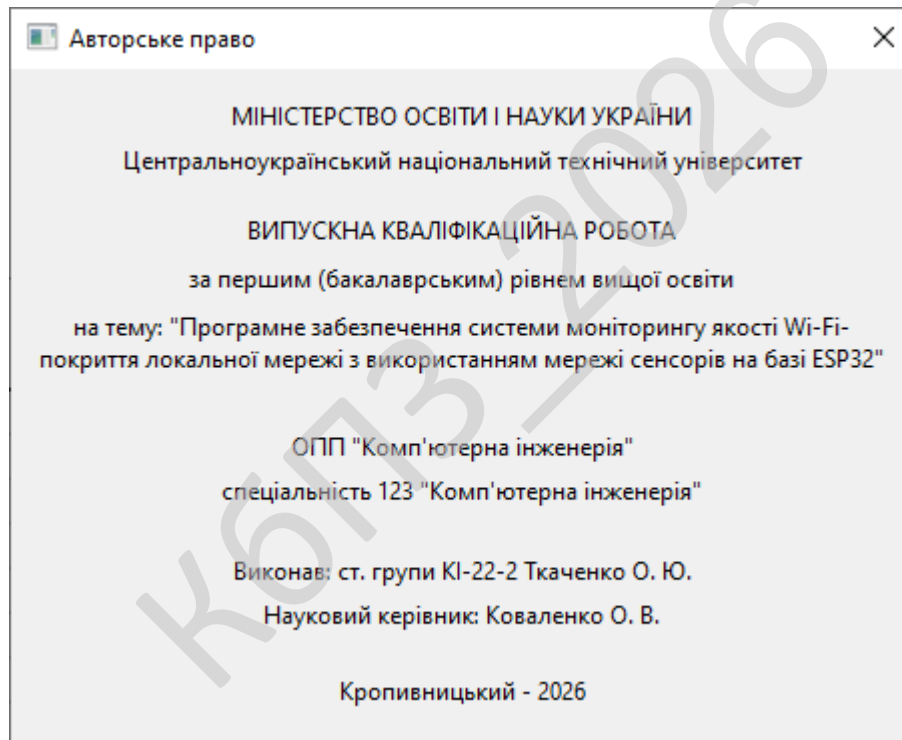


Рисунок 5.2 – Вікно авторського права

Для розгортання в реальному об'єкті достатньо розмістити поряд з виконуваним файлом програми файл mqtt_config.ini з адресою брокера, портом, логіном і паролем.

Розроблене програмне забезпечення поширюється за моделлю відкритого вихідного коду з ліцензією GNU GPL версії 3. Вибір обумовлений

технічними залежностями: Qt Community Edition, яка використовується для десктопного застосунку, вимагає від похідних робіт відповідності GPL. Прошивка вузлів на ESP32 використовує бібліотеки з різними ліцензіями: mbedTLS поширюється під Apache 2.0, PubSubClient під MIT, компоненти ESP-IDF під Apache 2.0. Всі вони сумісні з GPL, тому конфліктів немає.

Практично це означає, що будь-хто може завантажити, зібрати і модифікувати систему для власних потреб, але зобов'язаний відкрити вихідний код власних змін, якщо поширює модифікований продукт далі. Для академічного і некомерційного використання, на яке орієнтована ця розробка, така модель є оптимальною.

КБПЗ-2026

Змн	Арк	№ докум	Підпис	Дата

ВКРБ-123.26.0060.00.00.ПЗ

6 ОСНОВНІ ВИСНОВКИ

У ході виконання бакалаврської роботи я розробив програмне забезпечення для моніторингу якості Wi-Fi-покриття локальної мережі з використанням мережі сенсорів на базі ESP32. Система складається з трьох мікроконтролерів, одного майстер-вузла та двох вузлів-підлеглих, які обмінюються даними через MQTT-брокер. Десктопний застосунок на Qt отримує RSSI-значення від усіх вузлів і будує теплову карту покриття в режимі реального часу.

Реалізований механізм ініціалізації через BLE з використанням ECDH-обміну ключами та AES-128 шифруванням показав себе практично зручним. Вузли отримують облікові дані для Wi-Fi та MQTT без жодного ручного втручання після першого налаштування майстер-пристрою через веб-інтерфейс. Це суттєво спрощує розгортання системи в нових приміщеннях.

Однак головна проблема, з якою я зіткнувся протягом усієї роботи, це ненадійність визначення позицій вузлів через трилатерацію на основі RSSI. Сигнал Wi-Fi і BLE надзвичайно чутливий до перешкод, відбиттів від стін, меблів та людей у приміщенні. Навіть з реалізованим експоненційним згладжуванням та оптимізатором із 25 перезапусками, похибка в оцінці позиції вузла часто становила 1-3 клітинки сітки, що при масштабі 0.5 метра на клітинку дає до 1.5 метрів відхилення. У вузьких коридорах або кімнатах з великою кількістю металевих поверхонь точність падала ще більше.

Проблема не в алгоритмі оптимізації і не в кількості перезапусків. Проблема фундаментальна: RSSI не є надійною мірою відстані в умовах приміщення. Я перепробував різні значення коефіцієнта загасання шляху від 2.0 до 3.5, різні опорні значення потужності, додавав штрафи за напрямок та зміщення за осями, і все одно карта позицій вела себе нестабільно при зміні умов середовища. Два однакових RSSI-значення, виміряних в різний час у тій

Зми	Арк	№ докум	Підпис	Дата

ВКРБ-123.26.0060.00.00.ПЗ

Арк.

50

самій точці, можуть давати різницю в 5-10 дБм просто через те, що хтось пройшов поряд або зачинили двері.

Для зовнішніх розгортань, наприклад моніторинг Wi-Fi покриття на відкритій території або у великих промислових об'єктах з вікнами, GPS-підхід є практично безальтернативним. Кожен сенсорний вузол сам знає свою точну координату і передає її разом з RSSI-виміром. Десктопний застосунок у такому випадку просто малює реальну карту покриття без жодної оптимізації чи апроксимації позицій. Теплова карта стає достовірнішою, а час налаштування системи скорочується, бо зникає потреба вручну вказувати розташування точки доступу.

Реалізований застосунок тим не менш виконує покладені на нього функції в умовах, де позиції вузлів відомі або приблизно задані оператором вручну. Механізм накопичення теплових зразків по клітинках сітки та їх усереднення дає відносно стабільну картину розподілу сигналу при тривалій роботі системи. Збереження та завантаження теплових карт у форматі JSON дозволяє порівнювати стан покриття в різні моменти часу. Логування всіх вимірювань у CSV-файл із часовими мітками дає матеріал для подальшого аналізу.

З технічного боку система працює. Три вузли підключаються, обмінюються даними через MQTT, десктопний застосунок отримує виміри і будує теплову карту в реальному часі. Ініціалізація через BLE з шифруванням відпрацьовує без збоїв, логування пише коректні CSV-файли, збереження і завантаження стану карти функціонує так, як задумано. Я запустив систему в реальному приміщенні і побачив, як теплова карта поступово заповнюється даними і відображає реальний розподіл сигналу.

Зми	Арк	№ докум	Підпис	Дата

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Rappaport T. S. Wireless Communications: Principles and Practice (2nd Edition). 2nd ed. Prentice Hall PTR, 2001. 736 p.
2. Gast M. 802.11 Wireless Networks: The Definitive Guide, Second Edition (Definitive Guide). O'Reilly Media, Inc., 2005. 630 p.
3. Developing IoT Projects with ESP32: Automate Your Home or Business with Inexpensive Wi-Fi Devices. Packt Publishing, Limited, 2021. 470 p.
4. Hands-On Internet of Things with MQTT: Build Connected IoT Devices with Arduino and MQ Telemetry Transport. Packt Publishing, Limited, 2019. 350 p.
5. Karl H., Willig A. Protocols and Architectures for Wireless Sensor Networks. Wiley & Sons, Incorporated, John, 2007. 526 p.
6. Gupta A. The IoT Hacker's Handbook: A Practical Guide to Hacking the Internet of Things. Apress, 2019. 340 p.
7. Kovalenko O., Khudov H., Mynko P., Ikhsanov S., Diakonov O., Solomonenko Y., Drob Y., Kharun O., Cherkashyn S., Serdiuk O. «Development A Method For Determining The Coordinates Of Air Objects By Radars With The Additional Use Of Multilateration Technology» Eastern-European Journal of Enterprise Technologies Volume 5, 2021, Pages 6-16. Режим доступу: <https://www.scopus.com/record/display.uri?eid=2-s2.0-85119667497&origin=resultslist> (Scopus).
8. Serhii Pohasii, Serhii Yevseiev, Oleksandr Zhuchenko, Oleksandr Milov, Volodymyr Lysechko, Oleksandr Kovalenko, Maryna Kostyak, Andrii Volkov, Aleksandr Lezik, Vitalii Susukailo «Development of crypto-code constructs based on LDPC codes» Eastern-European Journal of Enterprise Technologies 2/9 (116), 2022, Pages 44-59. Режим доступу: <https://www.scopus.com/record/display.uri?eid=2-s2.0-85130069202&origin=resultslist> (Scopus).

9. Khudov H., Baranik O., Kovalenko O., Yakovenko Y., Chahan Y. «The Information Technology for Determining Vehicle Route Based on Ant Colony Algorithms» International Journal of Emerging Technology and Advanced Engineering, 2022, 12(12), Pages 117–128. Режим доступу: <https://www.scopus.com/record/display.uri?eid=2-s2.0-85130069202&origin=resultslist> (Scopus).

10. Hennadii Khudov, Volodymyr Bashynskiy, Oleksandr Kovalenko, Kristina Tahyan, Oleksii Fakadii « The methods for improving the quality of detection of inconspicuous aerial objects through the use of external radiation sources» International Journal of Emerging Technology and Advanced Engineering, 2023, Volume 13, Issue 3, Pages 91-100. Режим доступу: https://doi.org/10.46338/ijetae0323_09 (Закордонне фахове видання).

11. Hennadii Khudov, Oleksandr Kostianets, Oleksandr Kovalenko, Oleh Maslenko, Yuriy Solomonenko «Using software defined radio receivers for determining the coordinates of low-visible aerial objects» Eastern-European Journal of Enterprise Technologies Vol. 5 No. 9 (124), 2023, Pages 61-73. Режим доступу: <https://www.scopus.com/record/display.uri?eid=2-s2.0-85172343893&origin=resultslist> (Scopus).

12. Смірнова Т.В., Моторін Ю.Ю., Буравченко К.О., Бочуля Т.В., Коваленко О.В. «Вибір оптимальної технології побудови хмарної інформаційно-комунікаційної системи автоматизації виробничих процесів» Вимірювальна та обчислювальна техніка в технологічних процесах, № 1 (2022). С. 15-26. 2022. Режим доступу: <http://vottp.khmnu.edu.ua/index.php/vottp/article/view/30/36> (Фахове видання. Категорія «Б»)

13. Вінтенко Б.Ю., Смірнов О.А., Коваленко О.В., Смірнов С.А., Коваленко А.С. «Дослідження нормативних документів та галузевих стандартів розробки програмного забезпечення комп'ютерних систем управління АЕС, важливих для безпеки». Системи управління, навігації та

зв'язку, 2023, вип. 2(72), С. 170-178. Режим доступу:
<https://doi.org/10.26906/SUNZ.2023.2.170> (Фахове видання. Категорія «Б»)

14. Вінтенко Б.Ю., Миронець І.В., Смірнов О.А. Коваленко О.В., Смірнов С.А., Буравченко К.О., Якименко Н.М. «Дослідження інформаційного забезпечення та технологічних регламентів процесів керування критичною інфраструктурою енергоблоку АЕС з реактором типу ВВЕР-1000». Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка». 2024. № 1(25), С. 253–278. Режим доступу:
<https://doi.org/10.28925/2663-4023.2024.25.253278> (Фахове видання. Категорія «Б»)

15. Вінтенко Б.Ю., Смірнов О.А., Миронець І.В., Смірнова Т.В., Коваленко О.В., Мацуй А.М. «Модель шляхів отримання вхідних даних комп'ютерної інтелектуальної системи підтримки оперативного персоналу АЕС». Центральнотуристський науковий вісник. Технічні науки. 2025. Вип. 11(42), ч. II. С.52-62. Режим доступу:
[https://mapiea.kntu.kr.ua/pdf/11\(42\)_II/11\(42\)_II_2025.pdf](https://mapiea.kntu.kr.ua/pdf/11(42)_II/11(42)_II_2025.pdf) (Фахове видання. Категорія «Б»)

16. Скакун П.П., Коваленко А.С., Коваленко О.В., Ткачук Р.О. «Комплексний аналіз підходів до управління ІТ-проектами в Україні» The scientific heritage (Budapest, Hungary), 2025, Volume 166, Pages 88-92. Режим доступу: <https://doi.org/10.5281/zenodo.16763561> (Закордонне фахове видання, країна ОЕСР).

17. Вінтенко, Б.Ю., Миронець, І.В., Смірнов, О.А., Коваленко, О.В., Усік, П.С., Буравченко, К.О., Лисенко, І.А. «Логіко-структурна модель комп'ютерно-орієнтованої процедури системи підтримки оперативного персоналу АЕС». Кібербезпека: освіта, наука, техніка. 2025. Том 2 № 30. С. 413-427, 2025. <https://doi.org/10.28925/2663-4023.2025.30.984> Режим доступу: <https://www.csecurity.kubg.edu.ua/index.php/journal/article/view/984> (Фахове видання. Категорія «Б»)

Зми	Арк	№ докум	Підпис	Дата

18. Boudlal H., Serrhini M., Tahiri A. A comprehensive review of wifi sensing technologies: Tools, challenges and future research directions. AIP Conference Proceedings. 2023. Vol. 2814, No. 1. 030004. DOI: 10.1063/5.0148509.

19. Geier J. Designing and Deploying 802.11 Wireless Networks: A Practical Guide to Implementing 802.11n and 802.11ac Wireless Networks. 2nd ed. Indianapolis : Cisco Press, 2015. 528 p.

20. Understanding RSSI in Wi-Fi Networks. Oscium. URL: <https://www.oscium.com>

21. Кравчук С. О. та ін. Основи побудови безпроводових сенсорних мереж : навч. посіб. Київ : КПІ ім. Ігоря Сікорського, 2024. 313 с.

22. Лисенко О. І., Валуйський С. В. Безпроводові сенсорні мережі: принципи побудови та застосування. Київ : КПІ ім. Ігоря Сікорського, 2023. 280 с.

23. Design and Implementation of a Wireless Sensor Network Based on the ESP32 for IoT Applications. 2024 IEEE International Conference on Information and Communication Technologies (ICICT). 2024. DOI: 10.1109/ELMAR62909.2024.10694153

24. ESP32 Wi-Fi API Documentation. Espressif Systems. URL: <https://docs.espressif.com/projects/arduino-esp32/en/latest/api/wifi.html>

25. ESP32 WiFi Scan for Networks Example (Arduino IDE). DeepBlue Embedded. URL: <https://deepbluembedded.com/esp32-wifi-scan-networks-example-arduino-ide/>

26. ESP32-Driven Real-Time Indoor Localization Using Wi-Fi RSSI and a Simple Path Loss Model. IEEE Conference Publication. 2024.

27. A Dual-Antenna Mobile Robot Orientation Estimation System Based on Wi-Fi RSSI Fingerprinting. 2025 IEEE International Conference on Robotics and Automation (ICRA). 2025.

28. Network Monitoring and Alert System using ESP32 and BLE. CS Open CourseWare. 2025. URL: <https://ocw.cs.pub.ro/>

Змн	Анк	№ докум	Підпис	Дата

29. Molisch A. F. Wireless Communications. 2nd ed. Chichester : Wiley-IEEE Press, 2011. 839 p.
30. Sklar B. Digital Communications: Fundamentals and Applications. 2nd ed. Upper Saddle River : Prentice Hall PTR, 2001. 1079 p.
31. Stallings W. Wireless Communications and Networks. 2nd ed. Upper Saddle River : Pearson Education, 2005. 559 p.
32. Misra S., Woungang I., Misra S. C. Guide to Wireless Sensor Networks. London : Springer, 2009. 707 p.
33. Sohraby K., Znati T., Minoli D. Wireless Sensor Networks: Technology, Protocols, and Applications. Wiley & Sons, Incorporated, John, 2006. 328 p.
34. Yick J., Mukherjee B., Ghosal D. Wireless sensor network survey. Computer Networks. 2008. Vol. 52, No. 12. P. 2292–2330. DOI: 10.1016/j.comnet.2008.04.002
35. Akyildiz I. F., Su W., Sankarasubramaniam Y., Cayirci E. Wireless sensor networks: a survey. Computer Networks. 2002. Vol. 38, No. 4. P. 393–422. DOI: 10.1016/S1389-1286(01)00302-4
36. Tanenbaum A. S., Wetherall D. J. Computer Networks. 5th ed. Boston : Pearson, 2011. 933 p.
37. Kurose J. F., Ross K. W. Computer Networking: A Top-Down Approach. 8th ed. Hoboken : Pearson, 2020. 800 p.
38. Petrick A., O'Hara B. IEEE 802. 11 Handbook: A Designer's Companion. IEEE, 2011. 404 p.
39. Perahia E., Stacey R. Next Generation Wireless LANs: 802.11n and 802.11ac. 2nd ed. Cambridge : Cambridge University Press, 2013. 478 p.
40. Espressif Systems. ESP-IDF Programming Guide. URL: <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/>
41. Espressif Systems. ESP32 Series Datasheet. URL: <https://www.espressif.com/en/support/documents/technical-documents>

42. Schwartz M. Mobile Wireless Communications. Cambridge : Cambridge University Press, 2013. 470 p.

43. Seidel S. Y., Rappaport T. S. 914 MHz path loss prediction models for indoor wireless communications in multifloored buildings. IEEE Transactions on Antennas and Propagation. 1992. Vol. 40, No. 2. P. 207–217. DOI: 10.1109/8.127405

44. Hashemi H. The indoor radio propagation channel. Proceedings of the IEEE. 1993. Vol. 81, No. 7. P. 943–968. DOI: 10.1109/5.231342

45. Goldsmith A., Chua S.-G. Variable-rate variable-power MQAM for fading channels. IEEE Transactions on Communications. 1997. Vol. 45, No. 10. P. 1218–1230. DOI: 10.1109/26.634685

46. Bahl P., Padmanabhan V. N. RADAR: An in-building RF-based user location and tracking system. Proceedings IEEE INFOCOM 2000. 2000. Vol. 2. P. 775–784. DOI: 10.1109/INFCOM.2000.832252

47. He S., Chan S.-H. G. Wi-Fi fingerprint-based indoor positioning: Recent advances and comparisons. IEEE Communications Surveys & Tutorials. 2016. Vol. 18, No. 1. P. 466–490. DOI: 10.1109/COMST.2015.2464084

48. Light R. Mosquitto: server and client implementation of the MQTT protocol. Journal of Open Source Software. 2017. Vol. 2, No. 13. P. 265. DOI: 10.21105/joss.00265

49. Gezici S. A survey on wireless position estimation. Wireless Personal Communications. 2008. Vol. 44, No. 3. P. 263–282. DOI: 10.1007/s11277-007-9375-z

50. Mao G., Fidan B., Anderson B. D. O. Wireless sensor network localization techniques. Computer Networks. 2007. Vol. 51, No. 10. P. 2529–2553. DOI: 10.1016/j.comnet.2006.11.018

51. Faragher R., Harle R. Location fingerprinting with Bluetooth Low Energy beacons. IEEE Journal on Selected Areas in Communications. 2015. Vol. 33, No. 11. P. 2418–2428. DOI: 10.1109/JSAC.2015.2430281

52. Suroso D. J., Arifin M., Cherntanomwong P. Distance-based indoor localization using empirical path loss model and RSSI in wireless sensor networks. Journal of Robotics and Control. 2020. Vol. 1, No. 6. P. 199–207. DOI: 10.18196/jrc.1638

53. Jondhale S. R., Maheswar R., Lloret J. Received Signal Strength Based Target Localization and Tracking Using Wireless Sensor Networks. Cham : Springer, 2022. 205 p.

54. Liu H., Darabi H., Banerjee P., Liu J. Survey of wireless indoor positioning techniques and systems. IEEE Transactions on Systems, Man, and Cybernetics, Part C. 2007. Vol. 37, No. 6. P. 1067–1080. DOI: 10.1109/TSMCC.2007.905750

K6ПЗ-2026

Змн	Арк	№ докум	Підпис	Пата

БКРБ-123.26.0060.00.00.ПЗ