МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
Факультет кібербезпеки та програмної інженерії
Кафедра інженерії програмного забезпечення

ДОПУСТИТИ ДО ЗАХИСТУ

Завідувач випускової кафедри

_____

"___" _____ 2023 р

ДИПЛОМНИЙ ПРОЕКТ
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСНИКА ОСВІТНЬОГО СТУПЕНЯ
“МАГІСТР”

Тема: “ Методика та програмний засіб обліку енергоефективності в системах «Розумний будинок»”

Виконавець: Бовкун Ярослав Петрович

Керівниця: к.т.н. Волкогон Вікторія Олексіївна

Нормоконтролер: Трофимчук Вікторія Миколаївна

Київ 2023

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки та програмної інженерії

Кафедра інженерії програмного забезпечення

Освітній ступінь магістр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

Форма навчання заочна

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____

"___" _____ 2023 р

ЗАВДАННЯ

на виконання дипломного проекту студента

Бовкуна Ярослава Петровича

1. Тема проекту: «Методика та програмний засіб обліку енергоефективності в системах «Розумний будинок»» затверджена наказом ректора від 04.10.2023р. № 2034/ст

2. Термін виконання проекту: з 02.10.2023 р. до 31.12.2023 р.

3. Вихідні дані до проекту: розробити програмний засіб за допомогою середовища об’єктно-орієнтованого програмування IntelliJ IDEA.

4. Зміст пояснювальної записки:

1. Дослідження предметної області системи;
2. Аналіз методів обліку енергоефективності;
3. Технолігії та компоненти програмного засобу;
4. Реалізація прототипу програмного засобу;
5. Висновки;

Календарний план-графік

№п/п	Завдання	Термін виконання	Відмітка про виконання
1.	Розробка графіка роботи. Ознайомлення з постановкою задачі та вивчення літератури Написання 1 розділу, представлення керівнику	02.10.2023 – 08.10.2023	
2.	Написання 2 розділу, представлення керівнику	09.10.2023 – 22.10.2023	
3.	Написання 3 розділу, представлення керівнику	23.10.2023 – 05.11.2023	
4.	Написання 4 розділу, представлення керівнику	06.11.2023 – 19.11.2023	
5.	Редагування та друк пояснювальної записки, графічного матеріалу	20.11.2023 – 03.12.2023	
6.	Проходження нормо-контролю, перепліт пояснювальної записки. Отримання відгуку керівника. Підготовка презентації та тексту доповіді	04.12.2023 – 05.12.2023	
7.	Розробка тексту доповіді. Оформлення графічного матеріалу для презентації	06.12.2023 – 16.12.2023	
8.	Предзахист кваліфікаційної Роботи. Отримання рецензії	17.12.2023	
9.	Підготовка матеріалів для передачі секретарю ДЕК (ПЗ, ГМ, CD-R з електронними копіями ПЗ, ГМ, презентації, відгук керівника, рецензія, довідка про успішність, 2 папки, 2 конверта)	18.12.2023 – 24.12.2023	
10	Захист дипломного проекту перед ЕК	25.12.2023 – 31.12.2023	

Дата видачі завдання 02.10.2023

Керівниця дипломної роботи: к.т.н. Вікторія ВОЛКОГОН

Завдання прийняв до виконання: Ярослав БОВКУН

РЕФЕРАТ

Пояснювальна записка до дипломної роботи «Методика та програмний засіб обліку енергоефективності в системах «Розумний будинок»»: 100 сторінок, 29 рисунків, 1 таблиця, 57 використаних джерел, 1 додаток.

ПРОГРАМНИЙ ЗАСІБ, УПРАВЛІННЯ ПЕРЕФЕРІЙНИМИ ПРИСТОЯМИ, АВТОМАТИЗАЦІЯ БУДИНКУ, ЕКОНОМІЯ ЕНЕРГОРЕСУРСІВ, АЛГОРИТМИ ЗБОРУ ТА АНАЛІЗУ ДАНИХ, СИСТЕМА РОЗУМНОГО БУДИНКУ.

Об'єкт дослідження – програмний засіб обліку енергоефективності в системах «Розумний будинок».

Мета дипломної роботи– підвищення енергоефективності в системах «Розумний будинок» шляхом використання методик обліку енергоефективності.

Метод дослідження – розробка програмного засобу для керуванням енергоефективності в системі «Розумний будинок» з використанням методик обліку енергоефективності.

В процесі роботи, був зроблений глибокий аналіз методів обліку енергоефективності та їх використання в системах «Розумний будинок». В результаті чого було використано неінтрузивний підхід моніторингу та метод найближчих сусідів.

Результати роботи можуть бути використані при розробці програмних засобів призначених для побудови систем: розумний будинок, розумне місто, розумний сад; в усіх галузях народного господарства, де є потреба використання автоматизованих систем контролю пристроїв та аналізу обліку енергоефективності, автоматизації різноманітних процесів для комфорту користувачів, економії викопних ресурсів, виконання дій за розкладом.

Розробка та дослідження проводилися під управлінням ОС Windows 11. Розробка програми проводилася у середовищі JetBrains PyCharm 2023.3.1., на мові програмування Python.

ABSTRACT

Explanatory note to the thesis "Methodology and software tool for energy efficiency accounting in "Smart House" systems": 100 pages, 29 figures, 1 table, 57 used sources, 1 appendix.

SOFTWARE, MANAGEMENT OF PERIPHERAL DEVICES, AUTOMATION OF THE BUILDING, ECONOMY OF ENERGY RESOURCES, ALGORITHMS OF DATA COLLECTION AND ANALYSIS, SMART BUILDING SYSTEM.

The object of the study is a software tool for accounting for energy efficiency in the "Smart House" systems.

The aim of the thesis is to increase energy efficiency in the "Smart House" systems through the use of energy efficiency accounting methods.

The research method is the development of a software tool for managing energy efficiency in the "Smart House" system using energy efficiency accounting methods.

In the process of work, an in-depth analysis of energy efficiency accounting methods and their use in "Smart House" systems was made. As a result, a non-intrusive monitoring approach and the nearest neighbor method were used.

The results of the work can be used in the development of software designed for the construction of systems: a smart house, a smart city, a smart garden; in all sectors of the national economy, where there is a need to use automated systems for controlling devices and analyzing energy efficiency accounting, automating various processes for the comfort of users, saving fossil resources, and performing actions according to the schedule.

The development and research was carried out under Windows 11 operating system. The program was developed in the JetBrains PyCharm 2023.3.1 environment, in the Python programming language.

ЗМІСТ

ПЕРЕЛІК ПРИЙНЯТИХ СКОРОЧЕНЬ

ООП – об’єктно-орієнтовне програмування

ОС – операційна система

ЗМ – запис та мастеринг

HTTP – HyperText Transfer Protocol

CSS – Cascading Style Sheets

JS – JavaScript

API - Application Programming Interface

IoT - Internet of Things

CPS - Cyber-Physical Systems

ICT - Information and Communications Technologies

OSI - Open Systems Interconnection

M2M - Machine-to-machine M2H Machine-to-Human RPi Raspberry Pi

RL - Reinforcement Learning

WSN - Wireless Sensor Network

HVAC - Heating, Ventilation, and Air Conditioning

ARCADIA - Architecture Analysis and Design Integrated Approach

MBSE - Model-Based System Engineering

EMS - Energy Management Systems

MQTT - Message Queue Telemetry Transport

NIALM - Non- intrusive Appliance Load Monitoring

NILM - Non- intrusive Appliance Load Monitoring

AS-NIALM - Automated Setting Non- intrusive Appliance Load Monitoring

MS-NIALM - Sanual Setting Non- intrusive Appliance Load Monitoring

ILM - Intrusive Load Monitoring

ВСТУП

На тлі глобальної енергетичної кризи та зростання цін на енергоносії, питання енергоефективності набуває все більшої актуальності. Одним з перспективних напрямків у цій сфері є впровадження систем «Розумний будинок», які дозволяють автоматизувати процеси керування інженерними системами будівлі та оптимізувати споживання енергоресурсів. Проте існуючі на сьогодні методики обліку енергоефективності таких систем не дозволяють в повній мірі оцінити ефект від їх впровадження. Це зумовлює необхідність розробки спеціалізованих методик та програмних рішень для комплексного аналізу показників енергоефективності систем «Розумний будинок».

Мета дипломного проекту полягає в розробці комплексної методики та програмного забезпечення, що дозволить здійснювати точний облік енергоспоживання та надавати засоби для ефективного управління енергетичними ресурсами в системах розумний будинок.

У ході роботи буде вирішено ряд актуальних завдань, таких як аналіз існуючих стандартів та технологій у сфері розумного будинку, розробка алгоритмів обліку енергоспоживання, інтеграція з існуючими системами управління, а також створення інтерфейсу для зручного моніторингу та управління енергоефективністю.

Отримані результати дозволять підвищити рівень автоматизації та оптимізації енергоспоживання в розумних будинках, сприяючи зменшенню енергетичних витрат та впровадженню сталого розвитку в галузі житлового будівництва.

Системи "розумний будинок" можуть суттєво підвищити енергоефективність за рахунок оптимізації використання енергоресурсів та зменшення витрат енергії. За допомогою автоматичного регулювання освітлення та опалення залежно від присутності людей у приміщенні, рівня природного освітлення тощо. Це дозволяє уникати зайвих витрат енергії. Моніторинг і візуалізація споживання енергоресурсів у режимі реального часу, може оперативно виявляти проблемні місця та приймати заходи. Такі як автоматична оптимізація розподілу навантажень між різними джерелами енергії (електрика, газ, відновлювані джерела). Прогнозувати

споживання енергії з використанням технологій штучного інтелекту для оптимального планування. Автоматизувати процеси управління мікрокліматом та інженерними системами будівлі для підтримки комфортних умов з мінімальними витратами.

Об’єкт проектування – програмний засіб обліку енергоефективності в системах «Розумний будинок».

Мета дипломної роботи– підвищення енергоефективності в системах «Розумний будинок» шляхом використання методик обліку енергоефективності.

Метод дослідження – розробка програмного засобу для керуванням енергоефективності в системі «Розумний будинок» з використанням методик обліку енергоефективності.

Задачі проектування:

1. Провести аналіз предметної області.
2. Виконати аналіз методів обліку енергоефективності
3. Сформувати вимоги до програмного продукту.
4. Спроектувати та реалізувати програмний засіб обліку енергоефективності в системі «Розумний будинок».

Практична цінність проекту полягає у розробці нового програмного засобу, що дозволяє користувачам зберігати, обробляти та виконувати розрахунки енергоефективності споживання будинком, для економої ресурсів та коштів на їх придбання, за рахунок автоматизованого рішення з використання методів обліку енергоефективності.

Розробленою інформаційною системою можуть користуватися власники квартир, будинків, садівники для здобуття розрахунків та автоматичного їх використання системою розумного будинку, задля економії ресурсів та комфорту перебування у приміщені.

У ході роботи з програмним засобом, користувачі можуть налаштувати систему моніторингу енергоспоживання відповідно до потреб конкретного об'єкта, задати перелік датчиків та параметрів спостереження, сформувати та проаналізувати енергетичний профіль будівлі, визначити цільові показники ефективності й порівняти їх з фактично досягнутим рівнем, перевірити відповідність обраної системи вимірів та обліку національним і міжнародним стандартам. Також користувач має можливість оперативно переглядати та зберігати дані про фактичне енергоспоживання у всіх необхідних розрізах та за заданий період.

Система має забезпечувати гнучкість налаштувань та зручність використання кінцевим користувачем відповідно до конкретних потреб та завдань.

РОЗДІЛ 1.
ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ СИСТЕМИ

1.1. Актуальність розробки програмного засобу обліку енергоефективності в системах «Розумний будинок»

Більшість енергії, виробленої у світі, походить із викопного палива (нафти, газу та вугілля), і на неї припадає понад 80% загального світового виробництва первинної енергії, як показано на рисунку 1.1. Однак світові запаси енергії не є невичерпними (за нинішніх темпів споживання нафта вичерпається через 54 роки, газ – через 63 роки, а вугілля – через 112 років) і значною мірою сприяють глобальному потеплінню через парникові гази, що виділяються цими копалинами. Справді, згідно з даними опублікованими Міжнародним енергетичним агентством, світовий попит на енергію може зрости на 45% до 2030 року, зокрема через демографічний розвиток та індустріалізацію таких країн, як Китай та Індія, які одні лише мають понад 2 мільярди жителів.

Очікується, що споживання електроенергії зростатиме вдвічі швидше, ніж середнє споживання енергії. Для того, щоб зменшити цю залежність від енергії на основі викопного палива, яка значною мірою відповідає за викиди парникових газів, у всьому світі було запроваджено кілька ініціатив у цьому контексті. Ці ініціативи спрямовані на стимулювання енергоефективності, обмеження будівництва та використання найменш ефективних електростанцій, що працюють на вугіллі[34], скорочення викидів метану в нафтогазовому секторі, а також реформування субсидій на викопне паливо. Ці заходи сприятимуть розвитку відновлюваної енергетики, зображено на рисунку 1.2.

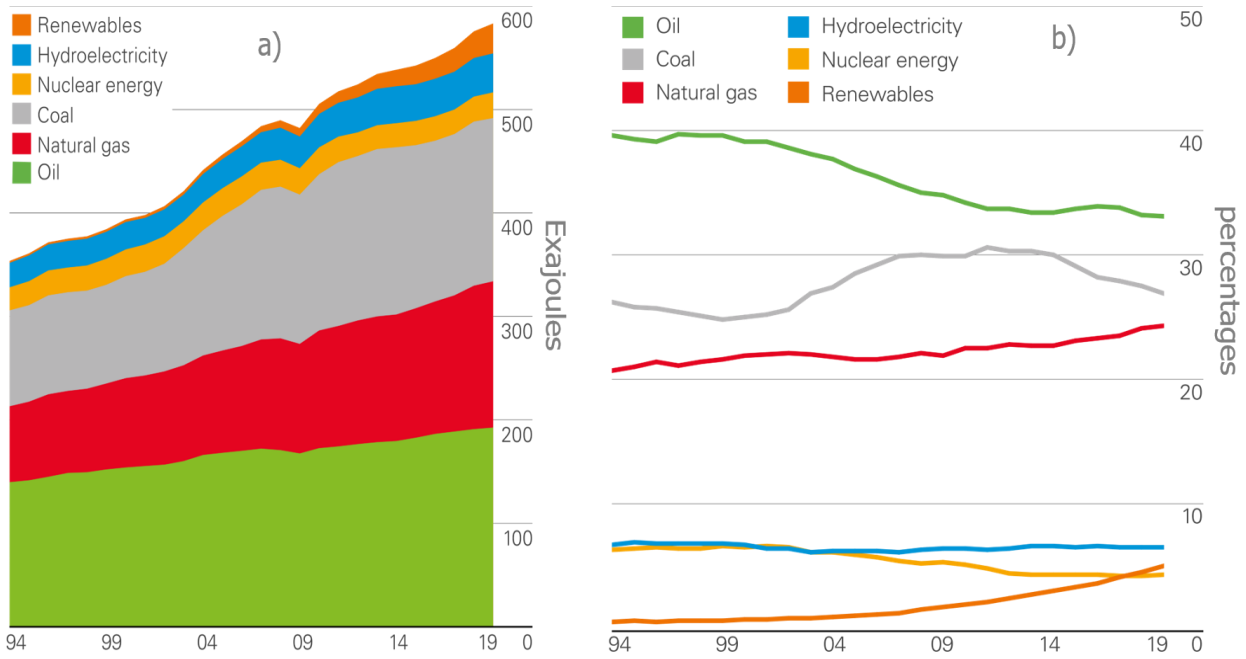


Рис. 1.1. Світове споживання первинної енергії, Частка світового споживання первинної енергії за паливом

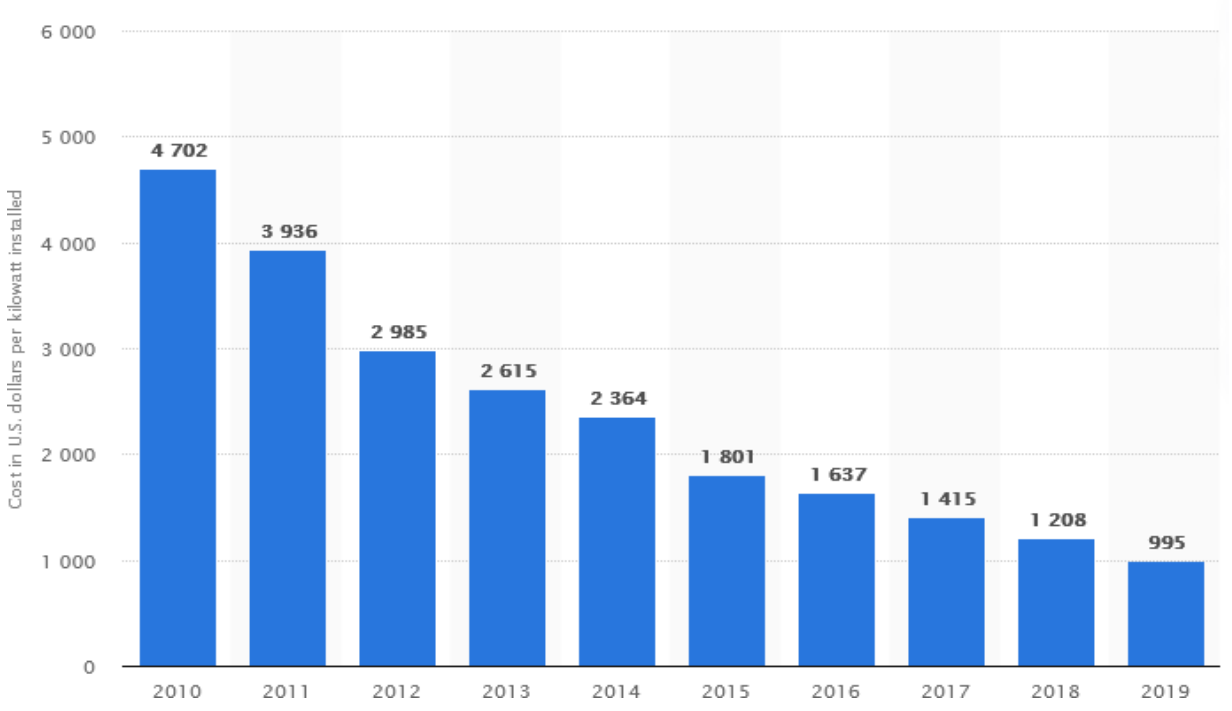


Рис. 1.2. Середня встановлена вартість сонячних фотоелектричних установок у всьому світі з 2010 по 2019 рік (у доларах США за кіловат)

Сьогодні ці відновлювані джерела енергії поступово стають окремими джерелами енергії, конкуруючи з викопним паливом з точки зору вартості та

продуктивності. Вони чисті та невичерпні, а тому здатні задовольнити наші енергетичні потреби в довгостроковій перспективі.

Серед цих відновлюваних джерел енергії сонячна енергія має великий потенціал. Більше того, зараз вона переживає потужний розвиток у світі. Такий розвиток подій можна пояснити урядовими заходами, спрямованими на скорочення використання викопного палива, і, з іншого боку, вражаючим падінням ціни на сонячні модулі та вартість встановлення сонячних панелей, які впали більш ніж на 90%.

Незважаючи на зростання споживання енергії, глобальні викиди вуглецю, пов’язані зі споживанням енергії, зросли на 0,5% у 2019 році порівняно із середнім показником на 1,1% на рік за останнє десятиліття[23]. Це незначне збільшення частково пояснюється підвищенням енергоефективності, а також кращим впровадженням нових технологій. Ці нові технології, можуть створити значний потенціал в енергетичному секторі. Як видно, дослідження ринку, опубліковане у 2015 році, передбачає збільшення з 42 до 155 мільярдів підключених об’єктів у всьому світі до 2025 року. Ці об’єкти можуть задовольняти потреби та надавати послуги. Як наслідок, вони стосуються різноманітних сфер застосування, таких як електронна охорона здоров’я, сільське господарство, навколишнє середовище, автомобільна промисловість, розумні міста та розумні будинки.

У цій дипломній роботі нас цікавить сфера розумних будинків. Розумний дім можна визначити як житло, обладнане пов’язаними об’єктами, які об’єднують комунікаційні та інформаційні системи для вимірювання, моніторингу та автоматичного контролю за споживанням енергії через підключення до Інтернету, за допомогою мобільного пристрою.

Використовуючи цю технологію, розумний дім також може підвищити безпеку та комфорт мешканців. Розумних будинків стає все більше, обладнаних системами відновлюваної енергії, такими як сонячні батареї, підключеними до загальної електричної мережі, для максимізації власного споживання та енергозбереження.

Для енергоефективних будівель завдання полягає в тому, щоб налаштувати ці об’єкти не лише для моніторингу споживання енергії, але й для управління енергією (зниження навантаження, коригування опалення, тощо) безпосередньо або через залучення мешканців. Мета цієї дипломної роботи полягає в тому, щоб запропонувати рішення для полегшення інтеграції, розгортання та взаємодії підключених об’єктів, пов’язаних бездротовою мережею всередині будинку, а також розробити управління живленням, яке б сприяло власному споживанню[18].

Це керування живленням прийматиме рішення відповідно до прогнозів спожитої та зібраної енергії. Очевидно, що ключовим принципом розумного будинку є збір відповідної інформації, а потім пропонування послуг, наприклад, пов’язаних з контролем та управлінням енергією.

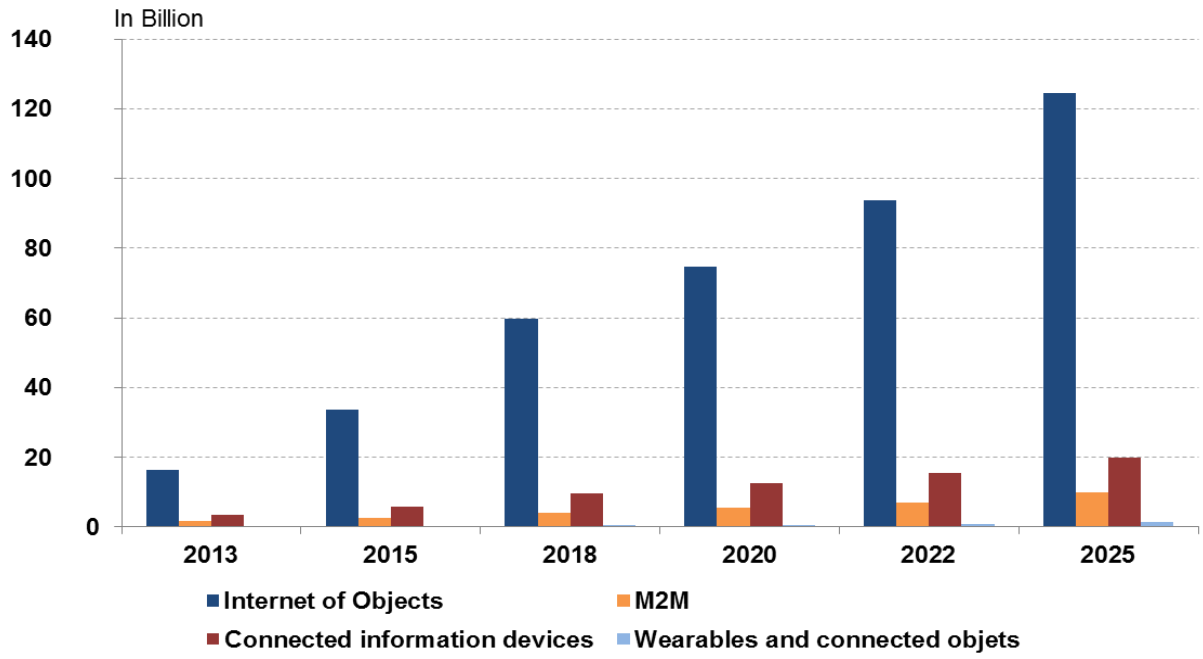


Рис. 1.3. Світовий ринок Інтернету речей, 2013 – 2025

Дійсно, житловий сектор є одним із головних споживачів енергії та становив 35 відсотків загального споживання енергії в Європі у 2018 році. Крім того, нещодавні дослідження показали, що постійний зворотній зв’язок (наприклад, через візуалізацію в реальному часі) у поєднанні з автоматизованим управлінням - система для домашнього обладнання може дозволити скоротити енергію від 15 до

30 відсотків. Тому інфраструктура збору даних, пов’язана з енергоспоживанням, є необхідним базовим елементом такої системи управління.

Сьогодні ідея розумного будинку полягає в тому, щоб об'єднати комфорт, захищеність, охорону здоров'я, безпеку та енергозбереження в будинку за допомогою комп'ютерів та Інтернету речей. Технологія, яку це надає, пропонує кращу якість життя для користувача, надаючи різні автоматизовані інструменти для кращого контролю вдома. Сьогодні використовуються системи віддаленого моніторингу, які дозволяють користувачеві керувати своїм будинком на відстані. Інші різноманітні функції, що використовують IoT, дозволяють користувачеві керувати своїм будинком та автоматизувати різноманітні пристрої для оптимізації комфорту користувача[51].

«Розумний будинок» – це житловий будинок сучасного типу, організований для проживання людей за допомогою автоматизації і високотехнологічних пристроїв. Під цим терміном слід розуміти систему, яка забезпечує безпеку та ресурсозбереження (в тому числі і комфорт) для всіх користувачів. Саме поняття було сформульовано Інститутом інтелектуальної будівлі у Вашингтоні (округ Колумбія) в 1970-х роках: це дім, що забезпечує продуктивне й ефективне використання робочого простору.

Зі збільшенням обчислювальної здатності гаджетів концепція «розумний будинок» отримала своє логічне продовження – систему «Інтернет речей», згідно з якою була проведена первинна стандартизація та визначені основні правила та рекомендації до побудови готового продукту на рівні як системи загалом, так і окремих компонентів. Незважаючи на відносну новизну, вже зараз існує декілька десятків різних рішень.

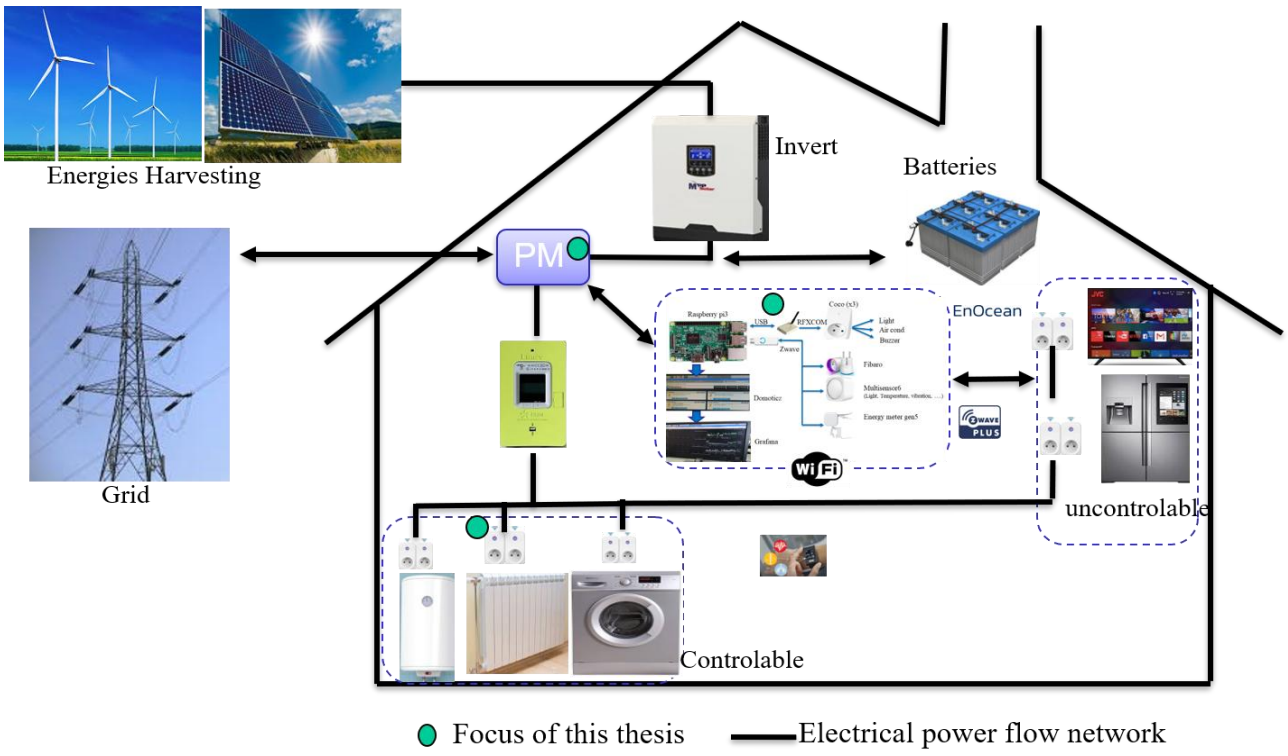


Рис. 1.4. Система домашнього енергоменеджменту

Системи домашньої автоматизації, безумовно привабливі, оскільки вони дозволяють контролювати все дистанційно через веб-додатки. Деякі додатки використовують найпростіші можливості, надані IoT (Інтернет речей), наприклад додатки для цілей безпеки (відеоспостереження, виявлення вторгнень, керування доступом), для управління та обслуговування підприємства (виявлення несправностей, управління), для автоматизації обслуговування і для розважальних. Порівняно з концепцією розумного урбанізму/розумного міста, концепція розумних будинків - як об’єкт дослідження в галузі досліджень інфраструктури. тобто як об’єкт, що складається з дому як простору, насиченого пристроями, підключеними до Інтернету[29]. Дослідження вітчизняних цифрових пристроїв більш чітко зосереджені на великих даних, цілісності, нагляді чи праці.

Ці простори вимагають дослідження інфраструктури програмного забезпечення. Як згадувалося на початку цього розділу, дослідження інфраструктури розумних міст і розумних технологій загалом зосереджені в основному на програмному забезпеченні, даних і наслідках, конфліктних інтересів у

обробці великих даних. Були також проведені дослідження звичайних пристроїв, що генерують дані.

«Розумний будинок» повинен вміти розпізнавати конкретні ситуації, що відбуваються в будівлі, і відповідним чином на них реагувати. Одна з систем може управляти поведінкою інших по заздалегідь виробленим алгоритмам. Основною особливістю інтелектуальної будівлі є об'єднання окремих підсистем в єдиний керований комплекс[51]. Важливою особливістю і властивістю «розумного будинку», яка відрізняє його від інших способів організації життєвого простору, є те, що це найбільш прогресивна концепція взаємодії людини з житловим простором, коли людина однією командою задає бажану обстановку, а вже автоматика відстежує режими роботи всіх інженерних систем і електроприладів. У цьому випадку виключається необхідність користуватися кількома пультами при перегляді телебачення, десятками вимикачів при управлінні освітленням, окремими блоками при управлінні вентиляційними і опалювальними системами, системами відеоспостереження та сигналізації, воротами і іншим. У smart-будинку достатньо одним натисканням на настінній клавіші або пульті дистанційного керування (ДК) чи сенсорній панелі, вибрати один зі сценаріїв, і будинок сам налаштує роботу всіх систем, відповідно до побажань господаря, часу доби, погоди, зовнішньої освітленості.

1.2.Що таке система енергоменеджменту в розумному будинку

Загалом, систему енергоменеджменту можна визначити як систему, яка дозволяє ефективно управляти енергією, як для споживачів (в нашому випадку «розумний дім»), так і для постачальників (мережі), зміщуючи попит на непікові ціни на енергію. Це може допомогти зберегти ресурси викопного палива та зменшити споживання енергії. У випадку розумного будинку система енергоменеджменту дозволяє спілкуватися, контролювати та дистанційно активувати побутову техніку, щоб не лише покращити якість життя та комфорт, а також оптимізувати власне споживання, таким чином зменшуючи рахунок за електроенергію. Розумні будинки — це повсюдна галузь обчислювальної техніки,

яка включає інтеграцію пов’язаних об’єктів у будинках для безпеки, комфорту, охорони здоров’я та енергоефективності. Ці пов’язані об’єкти є частиною поточних технологій в електромережах, які сприяють розгортанню інтелектуальних датчиків та інших передових вимірювальних пристроїв, які забезпечують зв’язок, моніторинг і дистанційне керування побутовими приладами. Розумні будинки пропонують кращу якість життя та більшу ефективність, використовуючи переваги дистанційного моніторингу та механізмів контекстної самоадаптації для визначення потреб і вподобань мешканців і координації роботи приладів. Дротові та бездротові мережі датчиків і приводів розгортаються в розумних будинках, дані датчиків збираються та зберігаються на локальній або віддаленій центральній платформі залежно від контексту програми. Ця платформа також відповідає за обробку отриманої інформації, що дозволяє оптимізувати керування та роботу побутових приладів в інтересах комфорту та енергоефективності мешканців.

1.3 Система розумного будинку

Кількість можливих систем і комбінацій пристроїв, що працюють разом у середовищі розумного будинку, необмежена. Тому пропонуємю конкретну систему щодо обладнання в розумному будинку: ми припускаємо середовище з різноманітними підключеними пристроями, включаючи смарт-телевізор, медіа-приставку, світлодіодні лампочки, жителя будинку з розумним годинником (такий пристрій може забезпечити важливою інформацією, головним чином про діяльність користувача), і датчик світла, встановлений на Raspberry Pi. У деяких з цих пристроїв присутні енергозберігаючі режими. Наприклад, екран смарт-годинника вимикається, якщо користувач не дивиться на годинник або не повертає зап’ястя, а телевізор вимикається або переходить у режим сну після періоду бездіяльності. У такому сценарії пристрої не обмінюються даними для досягнення більшої економії енергії, а енергоспоживання цих пристроїв невідоме, якщо вони не підключені до апаратного вимірювального обладнання.

Однак ми знаємо, що на потужність впливає багато показників (наприклад, на потужність телевізора впливає його яскравість, на потужність світлодіодної

лампочки впливає яскравість і колір, а на потужність медіа-приставки впливає декодування високоякісного потокового відео). Крім того, коли пристрій обмінюється даними, зібраними зі своїх датчиків, з усією системою, потужність може бути оцінена для кожного пристрою, і більш цікаві адаптації виконуються на низькому рівні, що веде до цілісної оптимізації енергії. Управління живленням виконується, забезпечуючи комфорт користувача: зменшення яскравості світлодіодів і телевізора, зниження якості потокового передавання, перевірки поведінки користувача та відповідні дії, а також оцінка та обмін енергоспоживанням пристрою.

1.3.1 Сучасні методи управління енергією

Є велика кількість існуючих досліджень управління енергією. Ці дослідження були ретельно переглянуті. Щоб краще проаналізувати методи керування живленням, були класифіковані таким чином:

Дизайн і моделювання: методи базуються на зниженні споживання енергії на етапах проектування пристрою чи системи. Це може бути використання нових технологій, внесення змін в архітектуру або дизайн системи, або за допомогою тренажерів вивчити поведінку компонента перед тим, як випустити його на ринок. Він спрямований на підвищення ефективності пристрою, одночасно зменшуючи його енергоспоживання.

Планування: полягає у зміні інтервалу часу, коли виконується певне завдання. Зміна часу виконання завдання, зазвичай, робиться для отримання вигоди від виробництва енергії, коли воно екологічніше, обирає кращий енергоефективний часовий проміжок виконання, або під час відсутності користувачів. Крім того, планування може мати на меті виявлення невикористаних пристроїв, для зміни їх режиму роботи. Зазвичай такий підхід реалізується в дата-центрах. Бездротова сенсорна мережа, розумні будинки, IoT.

Вибір користувача та рекомендації: поведінка користувача безпосередньо впливає на споживання енергії. Ці підходи засновані на заохоченні користувачів і рекомендації їм вживати заходів, які зменшують споживання енергії. Це також

включає підвищення обізнаності користувачів про їхнє споживання, шляхом показу їм візуальної інформації та сповіщень. Більшість розглянутих рішень фокусуються на індивідуальному рівні, рівень підприємства, та рівні громади. Крім того, інші використовували ігри, щоб показати користувачам своє енергоспоживання.

Адаптація та переналаштування: підходи засновані на здатності системи змінювати свою початкову конфігурацію шляхом налаштування деяких параметрів, що дозволяє їй адаптуватися до змін, які відбуваються під час виконання. Зазвичай він базується на кінцевій кількості випадків, які були попередньо визначені під час розробки або кілька попередньо визначених режимів роботи. Інші рішення пропонують автономні системи, які можуть використовувати методи машинного навчання та створювати бази знань.

Потужність/попит: попит на електроенергію регулярно змінюється з часом, як і енергопостачання, особливо з відновлюваними джерелами енергії, такими як фотоелектричні елементи та вітрові турбіни. Постачальники повинні узгоджувати виробництво електроенергії з попитом, щоб бути максимально ефективними. Використовується багато методів, наприклад прогнозування необхідної потужності у певний час на основі історичних записів, збалансування доступної електроенергії та попиту або накопичення енергії, коли вона доступна.

Міграція: методи зазвичай використовуються разом із віртуалізацією у великих розподілених центрах обробки даних у корпоративному середовищі. Віртуальні середовища можуть бути створені для легкої міграції програми з одного сервера на інший, менш завантажений[46]. Коротше кажучи, він заснований на зміні місця виконання завдання.

Щоб оцінити попередні рішення, було визначено кілька критеріїв порівняння, включаючи рівень автономності, гнучкість системи, загальне призначення підходу та неоднорідність. Ці критерії можна визначити таким чином:

Рівень автономності: п’ять рівнів зрілості автономії, визначені в системі автономних обчислень, використовуватиметься для класифікації пов’язаної роботи.

- Ручний рівень: користувачі виконують усі функції керування.

- Рівень моніторингу: дані з керованих ресурсів збираються, допомагаючи користувачам мінімізувати час, необхідний для збору та розуміння інформації.
- Рівень аналізу: технології використовуються для забезпечення кореляції між показниками. Він розпізнає шаблони, передбачає найкращу конфігурацію та пропонує користувачеві поради щодо можливих дій.
- Рівень замкнутого циклу: середовище може автоматично виконувати дії на основі наявної інформації та знань про те, що відбувається в середовищі.
- Замкнутий цикл із рівнем цілей: бізнес-політики високого рівня, орієнтовані на користувача, контролюють роботу інфраструктури в автоматизований спосіб.

Гнучкість системи: здатність системи працювати з динамічним середовищем. Це включає в себе: додавання, оновлення та видалення пристроїв або змін у компонентах програмного/апаратного забезпечення, а також у топології мережі.

Головна мета: визначає загальність або специфічність рішень, щодо потенційних областей застосування. Він дає можливість зрозуміти, чи можна легко застосувати підхід до будь-якої області та чи він все ще ефективний за різних обставин і різних пристроїв.

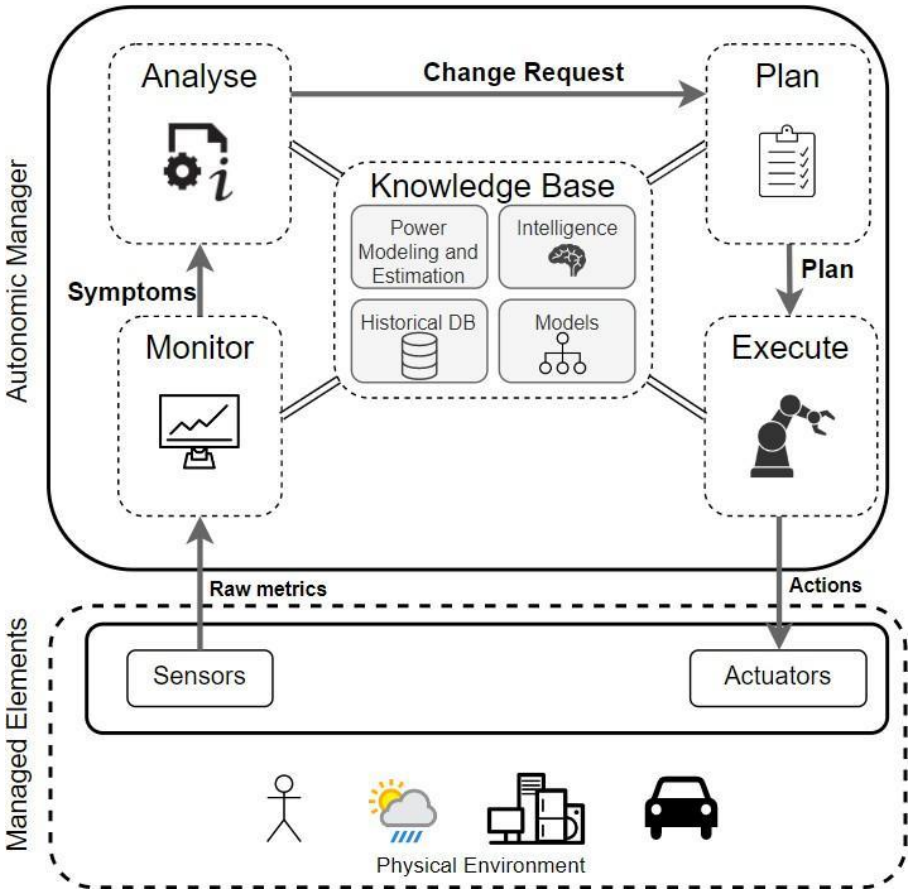
Контрольовані рівні та показники: структурні рівні, де розробляються рішення та збираються показники. Він забезпечує розуміння шарів, на яких досягається оптимізація, і чи обмежується вона цими шарами.

Неоднорідність: він визначається наявністю різноманітних апаратних і програмних інфраструктур, які використовують різні протоколи, компоненти та архітектури та які можуть надаватися різними постачальниками

Окрім відсутності гнучкості у вищезазначених роботах, впровадження проводилося на конкретних типах пристроїв. Таким чином, можна спостерігати відсутність неоднорідності. Ці недоліки спонукають до необхідності продовження дослідження з метою пошуку кращого підходу, який може розглянути вищезазначені проблеми, дотримуючись визначених критеріїв.

1.3.2 Архітектура автономних обчислень MAPE-K

Архітектура, заснована на підході до автономних обчислень MAPE-K (монітор, аналіз, план, виконання, база знань). автоматизований підхід до емпіричної генерації моделей оцінки потужності, для великого набору пристроїв і автоматизований генератор агента підкріплення навчання на основі знань, для вибору оптимальних дій виконання. Будь-які зміни або оновлення, які відбуваються в середовищі, вважаються такими, що призводять до калібрування моделей оцінювання та агентів навчання з підкріпленням. Прийняття підходу MAPE-K гарантує нефункціональні можливості, включаючи самоконтроль, самоконфігурацію, самооптимізацію, самовідновлення та самозахист[33]. Крім того, він гарантує функціональні можливості, що охоплюють збір даних, моніторинг,



аналіз, планування, виконання та приведення в дію. Наш підхід до архітектури поділяється на 5 основних компонентів, визначених таким чином:

Рис. 1.3: Архітектура MAPE-K

Монітор: збирає колекцію корисних даних з кожного з попередньо перерахованих рівнів, щоб ідентифікувати пов’язаний з енергією витік. Дані можна отримувати за допомогою інструментів оцінки програмного забезпечення, фізичних датчиків або інших засобів. У компоненті моніторингу дані очищаються, фільтруються та агрегуються. Зібрані дані можуть бути пов’язані з контекстом, зв’язком, обробкою, програмним забезпеченням або станом пристроїв. Він вибирає важливі ознаки, корелює та організовує їх як симптоми, які надсилаються в об’єкт аналізу.

Аналіз: це суб’єкт, відповідальний за використання та обробку зібраних даних, отриманих від монітора. Крім того, виявляє, що змушує систему споживати більше енергії, і що можна зробити, щоб зменшити це споживання на основі стратегій управління високого рівня. Він класифікує витрати енергії та обробляє їх для встановлення причини.

План: пропонує набір дій, яких необхідно виконати для досягнення мети високого рівня, пов’язаної зі споживанням енергії. Він відповідає за планування того, що слід змінити, щоб зменшити енергію системи, і надсилання цього сигналу зворотного зв’язку назад, для його виконання. Розробляються короткострокові та довгострокові плани. Вони сильно базуються на: запит на зміну, надісланий аналізатором і сховищем.

Виконання: отримує набір дій і надсилає їх виконавцям. Дії, які виконуються на пристрої, можуть бути поведінковими (наприклад, зміна частоти обміну даними) або структурними (наприклад, зміна режиму роботи пристрою на сплячий режим).

База знань: включає в себе моделі, для оцінки споживання енергії пристроями, модель онтології метрик (яка надає когнітивні можливості, які дозволяють знаходити зв’язок між метриками), споживання енергії та дій для запобігання, зображено на рисунку 1.4. Він також включає методи машинного навчання, які знаходять найбільш підходящу адаптацію, для мінімізації споживання енергії.

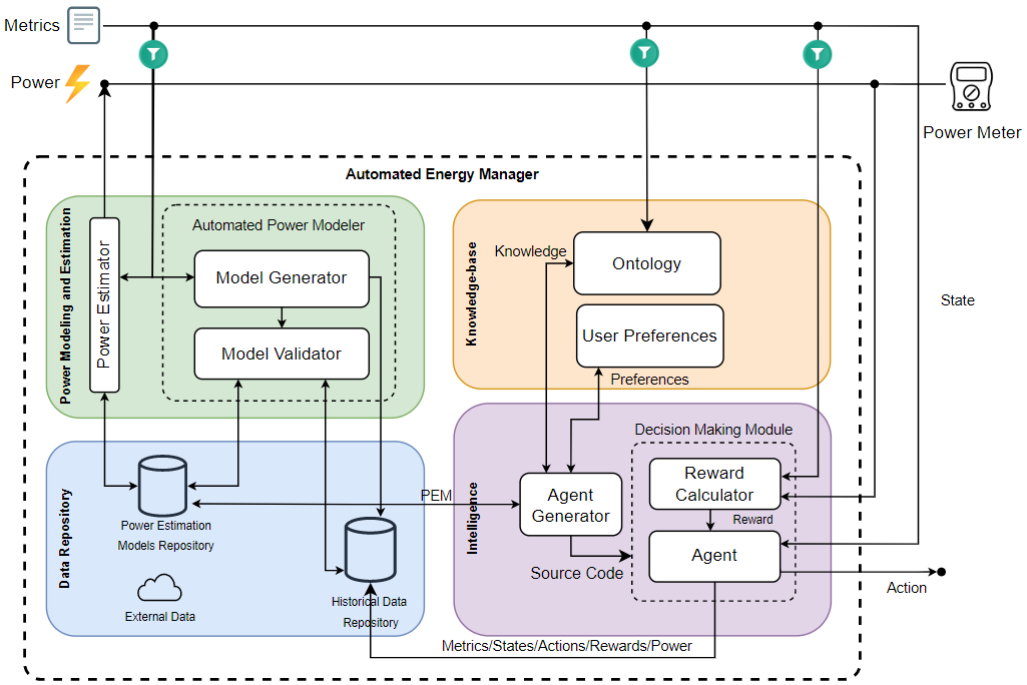


Рис. 1.4: Архітектура бази знань

Спільна база знань архітектури автономного обчислення, показує свою архітектуру, розділену на 4 основні компоненти:

Моделювання та оцінка потужності: автоматично оцінює енергоспоживання пристрою на основі зібраних показників без постійної потреби вимірювального обладнання.

База знань: використовується для розуміння зв'язку між показниками різних пристроїв, енергоспоживанням і діями, які можна застосувати. Ця модель співвідносить показники та потенційні дії, які має виконати будь-який пристрій, представлений у середовищі. Використання онтології гарантує включення різнорідних пристроїв і загальне призначення підходу.

Інтелект: відіграє центральну роль у визначенні найкращої енергоефективної дії, яку слід вжити, і її впливу на всю систему, враховуючи при цьому бажання користувачів. Він складається з генератора агентів, який створює агент на основі знань, отриманих від компонента бази знань, і моделі прийняття рішень, яка отримує показники енергоспоживання, щоб вибрати найкращу дію, яку можна вжити для економії енергії. Разом вони гарантують високу гнучкість і рівень автоматизації.

Репозиторій даних: колекція баз даних, яка включає раніше зібрані показники, застосовані дії, енергоспоживання, моделі оцінки потужності та потенційні зовнішні дані.

1.3. Архітектура підходу до енергоменеджменту

Поки ціни на електроенергію, викиди парникових газів і глобальна середня температура зростають, користувачі шукають рішення, щоб зменшити свій вплив на навколишнє середовище. З цієї точки зору можна досягти значного прогресу в управлінні різними та численними побутовими приладами. Розумний дім може бути можливим рішенням, для домашнього енергозбереження. Визначення приладів відіграє важливу роль у розумному домі, оскільки може забезпечити оптимізовані та автоматичні енергоефективні рішення.

Щодня в будинках встановлюють розумні лічильники, або пристрої моніторингу розумних розеток. Однак існуючі системи є дорогими і зазвичай складними у впровадженні та використанні. Нещодавні дослідження показали, що безперервний інформаційний зворотній зв’язок і точно налаштоване автоматизоване управління домашнім обладнанням, може дозволити скоротити рахунки за електроенергію з 15% до 30%. Пропонується все більше і більше мобільних, або веб-систем моніторингу енергії, які надають відповідну інформацію кінцевим користувачам.

Однак, щоб бути більш адаптивними та ефективними, вони повинні інтегрувати автоматичне розпізнавання запущених пристроїв. Така ідентифікація може бути здійснена за допомогою алгоритмів автоматичного навчання, застосованих до енергоспоживання пристроїв. Для цього кінцеві користувачі повинні встановити системи моніторингу на кожному пристрої та вручну позначити відповідні пристрої. Користувачі можуть візуалізувати енергоспоживання за категорією приладу та виконувати оптимізацію, активуючи, або програмуючи правила керування. Ця система розпізнавання також може представляти інші утиліти, такі як виявлення дефектів, локалізація приладів в офісах, або лікарнях.

Кожен день на ринку з'являються нові розумні розетки або розумні лічильники. перераховує чотири з цих вилок. Для збору інформації потрібна архітектура на основі розумних розеток. Далі датчики надсилаються на шлюз і можуть бути візуалізовані на інформаційній панелі.

Варто зазначити, що ці розумні лічильники забезпечують змінну кількість електричних сигнатур різних типів (наприклад, реальна потужність і реактивна потужність). У цьому розділі ми пропонуємо вивчити інфраструктуру, для збору ознак, які використовуватимуться для оцінки впливу кількості та типу функцій, що використовуються, для ідентифікації приладу. Більшість приладів мають потужність, близьку до нуля. Така поведінка відповідає періодам часу, коли прилади не використовуються, як правило, у режимі очікування або у вимкненому стані. Щоб врахувати таку поведінку, ми пропонуємо техніку, засновану на згладжуванні даних, щоб класифікувати окремі пристрої, включені в набір даних.

Архітектурний дизайн нашого підходу до управління енергією з використанням методології системної інженерії. Запропонована архітектура управління енергією гарантує, що наш підхід є цілісним і може керувати різноманітними пристроями в середовищі, що розвивається. Цей внесок дозволяє визначити функціональні та нефункціональні вимоги в спосіб, який можна відстежити, для належної перевірки наприкінці процесу розробки. Він документує всі рішення та вибір, зроблений під час розробки, логіку використання онтології та методів навчання з підкріпленням. Ми починаємо з операційного аналізу, що визначає потреби різних учасників і системи. Рівень архітектури системи має на меті визначити функціональні можливості, необхідні для задоволення операційних потреб, визначених на попередньому етапі. Третій крок зосереджений на логічній архітектурі, що створює детальну архітектуру системи на рівні компонентів. Нарешті, фізична архітектура спрямована на перехід від абстрактної логічної архітектури до представлення того, як система буде розроблена та побудована на рівні реалізації.

1.4. Відомі підходи енергомоніторингу в розумному будинку

Екосистемою називають управління розумним будинком з центрального пульта або віддаленого доступу.

Варіантів всього два:

1. Зібрати схему своїми руками, використовуючи різні хаби; для кожного пристрою вручну підібрати і налаштувати свій плагін; керувати за допомогою саморобних додатків.

2. Купити готовий стартовий набір, після чого придбати сумісні з ним пристрої, підключити їх до хабу, скачати спеціальний додаток на смартфон і керувати власнимінтелектуальним житлом.

У другому випадку користувач отримує готове рішення і гарантований результат. При цьому сума вкладень може бути цілком демократичною. Досить придбати базовий комплект, що включає в себе контроль безпеки при пожежі, витік води, несанкціоноване проникнення.

Сьогодні існує багато технології об'єднання і управління системами «розумногобудинку». Розглянемо деякі популярні готові рішення і проаналізуємо їх переваги і недоліки.

1) Ajax StarterKit (рисунок 1.5.). Український бренд Ajax представляє систему свого смарт будинку, із забезпеченням контролю безпеки. Ajax StarterKit попередить власника про несанкціоноване проникнення, спалах, затоплення. У комплект входять:

- центральний контролер;
- сирена;
- брелок з функцією пульта;
- сенсори положення дверей і вікон (відкрито/закрито);
- розбитого скла;
- протікання води;
- руху (розрізняє людей і тварин).

Рис. 1.5. Система «розумного будинку» Ajax StarterKit

Переваги:

- захищений радіоканал;
- простота налаштування і управління;
- швидке оповіщення;
- резервне живлення контролера. Недоліки: виключно охоронні функції. Середня ціна: 11000 грн.

2) Xiaomi. Відома китайська компанія, що випускає високоякісну електроніку. Їїекосистема, не обмежуючись базовим набором, виробляє безліч смарт пристроїв за доступними цінами. Це дозволяє організувати розумний будинок з широким функціоналом. Шлюз Мijia (рисунок 1.6.), поєднує в собі функції нічника, радіоприймача, хаба. Датчики вікон і дверей відстежують несанкціоноване проникнення, відправляють повідомлення власникові і включають камеру. При відкритті вікна вранці, автоматично включається освіжувач повітря. При відкриванні дверей вранці в спальню дитини вмикається приємна мелодія будильника. Бездротовий вимикач (бездротова кнопка) дозволяє одним натисканням розбудити дитину під час приготування сніданку на кухні. Йдучи на роботу, можна одним натисканням знеструмити всі електроприлади.

Рис. 1.6. Система «розумного будинку» Xiaomi Mijia

Інший важливий елемент системи – розумна розетка. Підключивши її до багатофункціонального шлюзу, використовуючи додаток на смартфоні, можна окремо запрограмувати включення і відключення будь-якого електроустаткування за часом доби. Для цього в додатку на кожен прилад налаштовуються свої іконки. Датчик руху вмикає світло в передпокої при відкриванні вхідних дверей, кондиціонер при відході до сну, запалить нічник або світлодіодну стрічку, якщовночі знадобитися відвідати туалет.

Весь набір коштує 6700 грн.Переваги:

- реалізація безлічі корисних функцій вже зі стартовим набором елементів;
- сумісність з Android і iPhon;
- все налаштоване «з коробки».Недоліки:
- китайські розетки не придатні для європейських вилок, потрібне придбання перехідника;
- шлюз в режимі радіоприймача приймає тільки китайські радіостанції.

3) Redmond (рисунок 1.7.). Продукція цього бренду відрізняється найширшим вибором пристроїв, керованих одним додатком. Голосове управління поки не реалізовано, алеза кількістю виробленого обладнання Redmond знаходиться

в лідерах. У комплект входять: розетка, датчик руху, датчик контролю положення дверей (герконовий) (рис. 2.3). Решта пристроїв, включаючи центр управління Redmond SkyCenter 11S, купуються окремо.

Рис. 1.7. Система «розумного будинку» RedmondB

Переваги:

- величезну кількість IoT пристроїв,
- можливість повної автоматизації будинку;
- якість та надійність.

Недоліки:

- відсутність голосового управління;
- для дистанційного керування екосистемою зі смартфона потрібна наявність смарт пристрою в будинку (планшет або смартфон).

4) Google. Американський бренд китайського виробництва. Його розумний будинок управляється і мобільним додатком, і голосом. Асистент відгукнеться на вітання «Окей Гугл» і виконає прохання включити музику, завести будильник, і т. д. В якості керуючого центру використовується музична колонка (єдиний пристрій, який виробляє Google). Всі інші гаджети розробляються фірмами-партнерами

гіганта

– Xiaomi, TP-Link або Phillips. Тому список сумісного обладнання досить великий. Середня ціна колонки Google Home – 5100 грн. Решта обладнання купується окремо, від інших виробників. Вартість комплекту, здатного регулювати освітлення, тепло, воду, управляти іншим домашнім обладнанням, в кожному окремому випадкувизначається індивідуально.

Переваги:

- наявність розумної колонки;
- можливість голосового управління;
- безліч сумісного з екосистемою домашнього обладнання від різних виробників.

Недоліки: висока ціна.

5) Amazon. Найближчий конкурент Google Home, американець китайського виробництва. Екосистема побудована навколо своєї смарт колонки, з якої чудово працюють IoT пристрою від різноманітних виробників. Це полегшує користувачам комплектацію необхідного набору, з можливістю подальшого розширення. Amazon вважається самим гнучким і перспективним брендом в цьому ряду, з урахуванням широкої сумісності, високої якості, маси новітніх розробок і помірних цін.

Так само як і Google Home, Amazon виробляє лише колонку, середня ціна якоїдорівнює 7350 грн. Решта обладнання комплектується іншими виробниками.

Переваги:

- гнучкість системи;
- можливість налаштування великої кількості сценаріїв автоматизації;
- сумісність з багатьма виробниками;
- голосове керування.

Недоліки: відсутність української мови.

Порівняльні характеристики різних екосистем для «розумного будинку» наведені в таблиці 1.1.

Таблиця 1.1.

Характеристики екосистем для «розумного будинку»

Екосистема	Ajax StarterKit	Xiaomi Mijia	Redmond	Google Home	Amazon
Простота налаштування	+	+	+	+	+
Відкритість системи	–	+	–	+	+
Мобільний додаток	+	+	–	+	+
Web-інтерфейс	+	+	+	+	+
Голосове керування	–	–	–	+	+
Базова вартість системи, грн.	10000	6700	7300	8000	9000

З розглянутих готових програмно-апаратних рішень найбільшим функціоналомі простотою інтеграції відрізняються Google Home і Amazon, оскільки вони елементарні в налаштуванні і встановленні додаткового обладнання. Але через високу ціну і необхідність «будувати» систему з пристроїв від різних виробників ці варіанти не є ідеальними.

На даний момент найпопулярніші пристрої розумного будинку використовуютьтехнології Wi-Fi, Bluetooth, ZigBee і Z-Wave. У кожної з технологій є свої плюсиі мінуси, і ніхто не забороняє використовувати їх разом, компенсуючи недоліки кожної. Але для різних завдань і різних типів розумних пристроїв використовуються різні технології. Наприклад, в побутовій техніці (телевізор, холодильник і кавоварка) зазвичай використовують Wi-Fi або Bluetooth, які також є в будь-якому телефоні. Причина – цією технікою користуються, навіть не маючи повноцінної системи «розумного будинку». Для автоматизації освітлення і клімату більше підходять модулі, що вбудовуються – ZigBee або Z-Wave, так як вони спеціально розроблені для інтеграції з існуючим світловим і кліматичним

устаткуванням. Але для їх повноцінної роботи потрібен спеціальний хаб.

1) **Wi-Fi.** Без Wi-Fi не обійтися в IP-камерах, телевізорах, аудіо/медіа-плеєрах і іншій техніці для передачі відеосигналу. Звичайно, Wi-Fi може використовуватися і в вимикачах світла, датчиках, термостатах, але відсутність ретрансляції сигналу і високе енергоспоживання не дозволяють робити на ньому датчики, що працюють роками. Кожен виробник для свого Wi-Fi-пристрою, будь то розумна лампочка, чайник, холодильник або робот-пилосос, випускає свою власну програму, і немає єдиного стандарту, щоб управляти всією технікою з однієї програми. Це не дозволяє зробити «розумний будинок» тільки на Wi-Fi по-справжньому зручним.

2) **Bluetooth.** Актуальна версія Bluetooth Low Energy 4.2 має мале енергоспоживання, завдяки цьому працюють крихітні бездротові навушники, колонки і різні датчики на батарейках. Проблеми тут ті ж, що і з Wi-Fi: відсутність загального стандарту управління змушує кожного виробника робити власний додаток, що незручно для користувача. Дуже важлива для розумного будинку технологія Mesh (чарункова мережа) з'явилася лише в версії 5.0, яка ще мало де використовується, але, можливо, майбутнє розумних будинків саме за Bluetooth LE 5.

3) **ZigBee.** ZigBee спочатку розроблявся для застосування в мережах з датчиків, таких як лічильники електроенергії, води, газу, датчики температури. Топологія мережі може бути різна, в тому числі комірчаста (mesh). Це означає, що будь-який датчик бачить всі інші датчики і може передавати сигнал через них, тобто використовувати ретрансляцію, що сильно збільшує надійність передачі. У 2007 році з'явився стандарт команд для управління «розумним будинком», так званий профіль «домашньої автоматизації». З ZigBee випускають майже всі пристрої для створення домашньої автоматизації: реле, димери, лампи, термостати, замки, датчики. Але побутових приладів типу холодильників і телевізорів з ZigBee немає. У порівнянні з іншими протоколами для «розумного будинку», у ZigBee-пристроїв найпривабливіші ціни, однак відсутність 100 % сумісності між пристроями і хабами різних виробників не дозволяє зібрати розумний будинок тільки на ZigBee.

Z-Wave. Бездротовий протокол, що розробляється з 2001 року спеціально для домашньої автоматизації. Головною його перевагою є повна сумісність між пристроями різних виробників. Датчик руху від Fibaro може управляти димером Qubino, а вся автоматизація при цьому базуватися на контролері RaZberry від Z-Wave.Me. На даний момент продається більше 3000 різних Z-Wave пристроїв, які покривають всі потреби «розумного будинку». Це найпопулярніший протокол для об'єктів площею від 10 до 500 м². У Z-Wave, так само як і в ZigBee, використовуєтьсятопологія mesh з підтримкою ретрансляції сигналу і автоматичним знаходженням кращого маршруту. Головний недолік – ціна. В середньому, вартість пристрою становить 60-80 євро, що приблизно вдвічі вище, ніж у аналогів з ZigBee.

Висновок

Зважаючи на описані у першому розділі обставини з розробки програмного засобу, а саме системи «Розумний будинок» з використанням методик обілку енергоефективності, яка дозволить зменшити витрати енергоресурсів та збільшити комфортне використання своїм будинком для звичайних користувачів.

Аналіз існуючих систем розумного дому показав, що при розробці системи доцільно використовувати перевірені методи проектування, уважно вивчати рекомендації до розробки систем і проводити тестування на реально побудованій системі розумних пристроїв.

Було визначено, що програмний засіб системи «Розумний будинок » є бажаним елементом в теперішній час. Система дає змогу економити ресурси планети, збільшити комфорт для мешканців квартир та будинків.

Для забезпечення якості використання інформаційної системи, необхідно проводити тестування застосунку на великій кількості клієнтів.

Обравши напрям роботи, я переходжу до аналізу методів обліку енергоефективності, формування чітких та статичних вимог для створення такого застосунку.

РОЗДІЛ 2
АНАЛІЗ МЕТОДІВ ОБЛІКУ ЕНЕРГОЕФЕКТИВНОСТІ

2.1. Підхід енергомоніторингу в розумному будинку

У літературі ми часто зустрічаємо два типи підходу до моніторингу споживання електроенергії побутовими приладами (рисунок 2.1.): неінтрузивний (NILM) або інтрузивний (ILM)

Рис. 2.1. Підхід, який використовується в розумному будинку для моніторингу приладів

Ненав'язливий підхід полягає в аналізі тимчасової та стабільної сигнатури, що з’являється, коли прилади вмикаються чи вимикаються. Цей підхід вимагає одного датчика з високою частотою дискретизації, розміщеного на вході в будинок. Він використовується для ідентифікації кожного пристрою, підключеного до мережі, на основі загального споживання. Відповідно процес ідентифікації пристрою виконується в 5 кроків

- **Збір даних**: система NIALM збирає значення з датчиків струму та напруги з визначеною частотою дискретизації.
- **Попередня обробка**: це важливий крок для фільтрації електричного шуму та отримання електричних характеристик, таких як загальна активна потужність,

реактивна потужність і повна потужність. Він може містити деякі складні завдання, такі як обчислення фази електричного сигналу, даних про гармоніки та коефіцієнта потужності. Чим більше електричних характеристик витягується, тим точнішою є покрокова класифікація.

- **Виявлення подій** працює над виявленням сукупних змін струму або потужності, щоб визначити подію для ввімкнення або вимкнення пристрою в електромережі.
- **Класифікація** групує події після етапу виявлення подій і зіставляє події ввімкнення/вимкнення для класифікації пристроїв.
- **Оцінка** - це останній процес підсумовування загального енергоспоживання та коефіцієнтів енергоспоживання для кожного пристрою.

2.1.1. Система енергетичного моніторингу на основі інтрузивних підходів (ILM)

Системи енергомоніторингу можуть впливати на поведінку мешканців, інформуючи їх через графічний інтерфейс про споживання електроенергії вдома в реальному часі. Якщо енергоспоживання кожної побутової техніки та споживчої електроніки з розбивкою відображається на настінному планшеті, комп'ютері чи телевізорі, мешканці знають про споживання електроенергії та можуть докласти зусиль, щоб зменшити споживання енергії. Протягом останніх років енергетика запропонувала веб-системи моніторингу та

Зверніть увагу, що вимірювання проводяться на кількох незалежних пристроях, щоб визначити внесок кожного пристрою. У результаті вартість високочастотного аналізатора, який використовується для виявлення перехідних періодів, була визнана недоліком. Хоча результати ідентифікації багатообіцяючі, деякі дослідники зосередилися на використанні аналізатора з частотою дискретизації до 15 кГц, щоб зменшити вартість. Незважаючи на легкість встановлення та налаштування системи збору даних, ненав'язливий підхід має недолік, пов'язаний із дефектами неточності (пов'язаними з накопиченням

споживання електроенергії різними пристроями, які створюють перешкоди в мережі). Інтрузивний підхід ILM, з іншого боку, полягає в аналізі в короткостроковій та середньостроковій перспективі (від кількох секунд до кількох хвилин) еволюції енергоспоживання одного чи кількох пристроїв, які використовуються або неактивні. Це нав'язливий процес, оскільки інтелектуальний лічильник (підключена розетка) розміщується на пристрої, який вимірюється. Цей підхід також має кілька переваг перед ненав'язливим підходом. Зібрана інформація точніша, а електричні сигнатури численні завдяки численним датчикам, які використовуються в цьому підході, що робить процес ідентифікації простішим і гнучкішим. Крім того, інтрузивний підхід дозволяє користувачеві дистанційно керувати приладами управління постачальників, щоб дозволити користувачам переглядати дані про домашню енергію та дистанційно керувати домашніми пристроями через Інтернет.

2.1.2. Система енергетичного моніторингу на основі неінтрузивнихпідходів (NILM)

Етап навчання автоматичної системи розпізнавання часто вимагає кількох серій заходів для врахування поведінки споживання електроенергії зарядами. Ці фази вимірювання виявляють кілька ступенів вторгнення. По-перше, вони вимагають вторгнення спеціаліста в помешкання, щоб встановити вимірювальні прилади на кожен прилад для збору різних підписів. Після першого кроку виявлення зміни стану другий крок полягає в ідентифікації навантажень за допомогою підписів пристроїв, що зберігаються в базі даних. Ці підписи можна отримати за допомогою різних методів, таких як генетичні алгоритми, методи цілочисельного програмування, або ймовірнісні приховані марковські моделі. Методи NIALM можна класифікувати на дві категорії:

— тих, хто використовує фазу автоматизованого навчання (AS-NIALM: автоматичне налаштування)

— ті, хто використовує фазу ручного навчання (MS-NIALM: ручне налаштування)

Ручне навчання: Ручні методи NIALM точніші, ніж автоматичні методи NIALM, завдяки збору сигнатур пристроїв, присутніх в установці. Однак це може бути дратівливим для клієнта і неприваблиле для дистриб'ютора або постачальника електроенергії. У випадку методів MS-NIALM створюється бібліотека сигнатур пристроїв на основі інтрузивних вимірювань на установці. Час відстеження пристроїв вимагає певної тривалості. Протягом цього періоду спостерігаються сигнатури пристроїв (активна потужність, реактивна потужність, середньоквадратичне значення струму тощо) та вручну кваліфікуються як сигнатури опалення, освітлення, прання тощо. - під час введення в експлуатацію та виведення з експлуатації.

Автоматичне навчання: У випадку методів AS-NIALM ознаки пристроїв покращуються з часом і ґрунтуються на інформації, зібраній в лабораторіях, а також зібраних у середовищах існування. Ці бібліотеки фактично ідентифікують кожен електричний пристрій та його споживання, і заохочують рух до ідентифікації використання цих пристроїв або навіть звичок використання або споживання останніх клієнтами. Методи MS-NIALM є інструментом у розробці методів AS-NIALM. Ймовірно, вони використовувалися для аналізу ситуацій, коли методи AS-NALM виявилися невдалими. Враховуючи їх набагато менш нав’язливий характер, методи AS-NIALM повинні тоді домінувати в більшості додатків шляхом створення власної бібліотеки підписів, яка спостерігається під час кроків зміни стану.

2.2. Управління приладами

Навантаження в розумному будинку можна розділити на дві категорії: неконтрольовані та контрольовані. Контрольовані навантаження визначаються як навантаження, які можна контролювати (наприклад, зміщувати в часі), не помічаючи впливу на спосіб життя споживачів. Керовані прилади постачальники все частіше пропонують і вже включають в кондиціонер, водонагрівач, сушильну машину і посудомийну машину. Неконтрольовані пристрої, які або дуже важливі, або можуть вплинути на спосіб життя споживачів при зміні часу споживання

електроенергії. Він включає в себе холодильник, пристрої приготування їжі та освітлення.

2.3. Методи машиного навчання в системах «Розумний будинок»

Контрольоване машинне навчання є одним із найбільш часто використовуваних і успішних типів машинного навчання. У цьому розділі ми представляємо деякі популярні алгоритми навчання, які використовуються в цій роботі. Контрольоване навчання використовується для прогнозування певного виходу з даного введення, і ми маємо приклади пар вхід/вихід. Ми створюємо модель машинного навчання з цих пар введення/виведення, які складають набір для навчання. Мета полягає в тому, щоб зробити точні прогнози для нових, ніколи раніше не бачених даних. Контрольоване навчання часто вимагає людських зусиль для створення навчального набору, але потім автоматизує та часто прискорює трудомістке або нездійсненне завдання.

Існує два основних типи проблем машинного навчання під наглядом, які називаються класифікацією та регресією. У класифікації метою є передбачити мітку класу, яка є вибором із попередньо визначеного списку можливостей. Класифікацію іноді поділяють на бінарну класифікацію, яка є окремим випадком розрізнення точно двох класів, і багатокласову класифікацію, яка є класифікацією між більш ніж двома класами. Для задач регресії мета полягає в тому, щоб передбачити дійсне число. Прогнозування споживання енергії протягом тижнів у будинку з огляду на вік та місце проживання мешканців є прикладом регресійного завдання. Під час прогнозування доходу прогнозована вартість – це сума, яка може бути будь-яким числом у заданому діапазоні.

Іншим прикладом завдання регресії є прогнозування врожайності кукурудзяної ферми з урахуванням таких атрибутів, як попередні врожаї, погода та кількість працівників, які працюють на фермі. Вихід знову може бути довільним числом. У наступних розділах представлено різні методи машинного навчання, які використовуються в цій дипломній роботі.

2.3.1. Алгоритм найближчих сусідів

Алгоритм K найближчих сусідів (K-NN) — це непараметричний контрольований алгоритм навчання, інтуїтивно зрозумілий і простий у реалізації. Цей алгоритм часто називають «лінивим навчанням» або на основі пам’яті, оскільки він базується лише на даних навчання. Його можна використовувати як для класифікації, так і для регресії. Його принцип полягає в обчисленні відстані між новими даними, що підлягають класифікації, та даними, на які посилаються, які формують навчальну базу даних. Невидимі дані класифікуються голосуванням за множиною своїх сусідів, при цьому дані призначаються найпоширенішому класу серед k найближчих сусідів. Існують різні типи обчислення відстані: Евклідова, Манхеттенська, Мінковського або Чебишева. Приклад класифікації за K-NN наведено на рисунку 2.2.

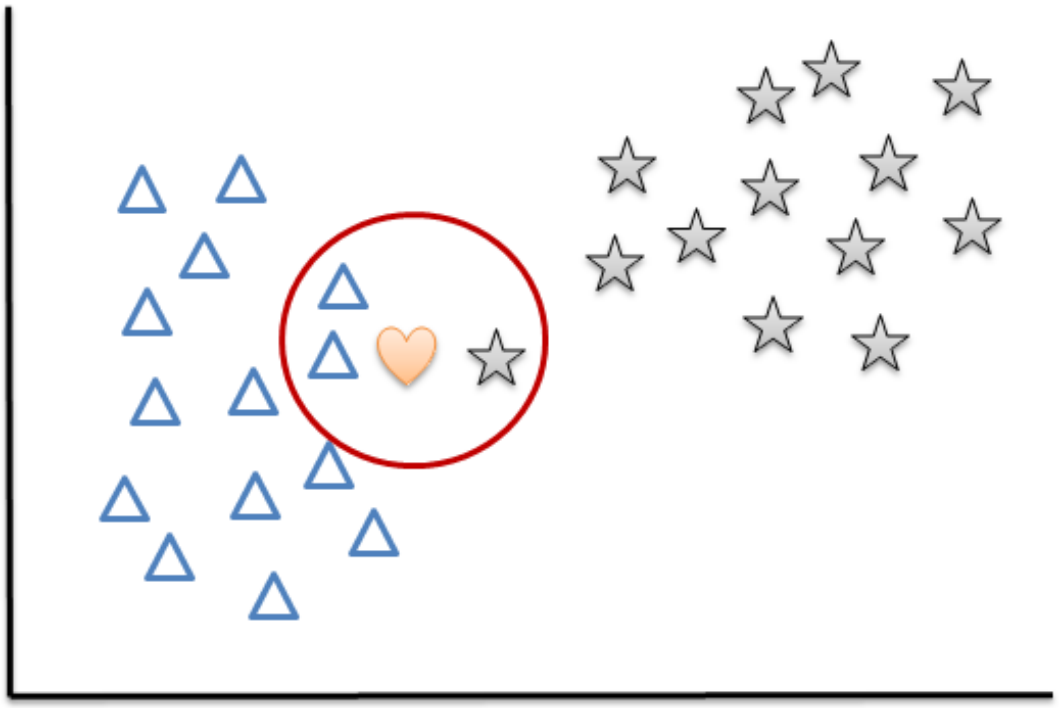


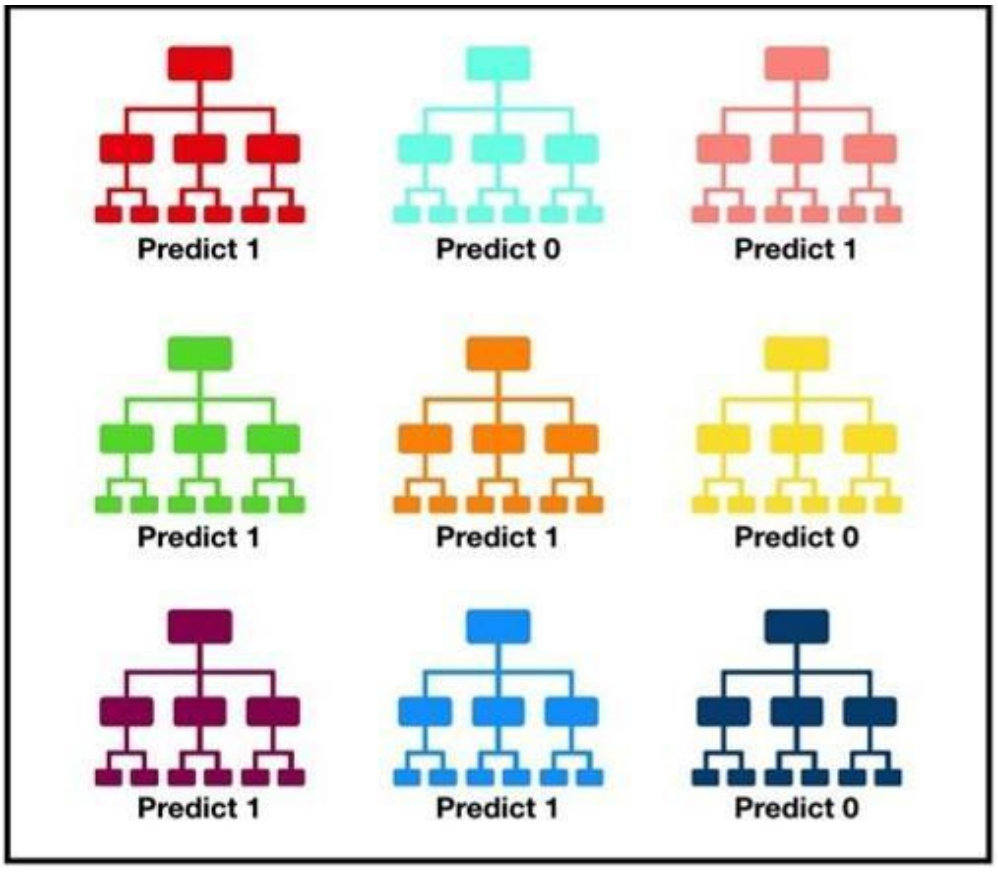
Рис. 2.2. Приклад класифікації 3NN

На цьому малюнку перший клас представлено трикутниками, а другий – зірками. Новий об'єкт для класифікації має форму маленького серця. Використовуючи класифікатор 3NN (k=3), ми знаходимо 2 найближчих сусідів

нового об’єкта, що належить до класу трикутників, проти одного з класів зірок. Тоді класифікатор вважає, що цей новий об’єкт належить до класу трикутників.

2.3.2. Метод випадкового лісу

Випадковий ліс в основному складається з великої кількості дерев рішень, які функціонують як набір. Кожне дерево трохи відрізняється від інших, представлено на рисунку 2.3. Ідея випадкових лісів полягає в тому, щоб зробити прийнятну роботу прогнозування для кожного дерева, але, ймовірно, буде переобладнано на деяких даних. Якщо ви будуєте багато дерев, усі з яких добре працюють і переобладнуються по-різному, ви можете зменшити кількість переобладнання,



усереднивши їхні результати.

Рис. 2.3. Візуалізація моделі випадкового лісу

Для реалізації цієї стратегії (зменшення переобладнання) нам доведеться побудувати досить велику кількість дерев рішень. Кожне дерево має виконувати прийнятну роботу з передбачення цілі, а також має відрізнятися від інших дерев. Випадкові ліси отримали свою назву завдяки впровадженню випадковості в структуру дерева, щоб гарантувати, що кожне дерево відрізняється. Є два способи рандомізації дерев у випадковому лісі: вибором точок даних, які використовуються для побудови дерева, і вибором функцій у кожному спліт-тесті. Щоб зробити прогноз за допомогою випадкового лісу, алгоритм спочатку робить прогноз для кожного дерева в лісі. Для регресії ми можемо усереднити ці результати, щоб отримати наш остаточний прогноз. Для класифікації використовується стратегія «м'якого голосування». Це означає, що кожен алгоритм виконує «м'яке» передбачення, надаючи ймовірність для кожного можливого вихідного тегу. Імовірності, передбачені всіма деревами, усереднюються, і прогнозується клас з найвищою ймовірністю.

2.4. Штучна нейронна мережа в системі «Розумний дім»

Одним із головних завдань системи енергоменеджменту в домогосподарстві є прогнозування та класифікація електричних сигнатур. Це завдання можна виконати за допомогою різних інструментів, таких як розумні розетки, а також інструментів візуалізації в реальному часі. Коли в побутовому приладі виникає аномалія (наприклад, несподіване надмірне споживання або відключення електроенергії), інформація змінюється, і користувачі повинні визначити свої аномалії відповідно до типу архітектури (IALM або NIALM), встановленої в будинку. Як було продемонстровано, штучні нейронні мережі є добре адаптованими інструментами, які допомагають користувачам краще керувати споживанням енергії, а також виявляти аномалії (тобто шляхом ретельного моніторингу приладу, який споживає скільки енергії порівняно зі звичайним використанням) у випадку розумного будинку. Штучну нейронну мережу (ШНМ) можна визначити як набір невеликих обчислювальних одиниць, з'єднаних між собою лініями зв'язку. Інформація, що

передається цими з’єднаннями, є цифровою і може бути закодована різними способами.

Кожен блок, який може мати невеликий обсяг локальної пам'яті, виконує обчислення на основі даних своїх з'єднань і локальних даних. Деякі нейронні мережі моделюють біологічні нейронні мережі, інші ні. Історично головною метою дослідження нейронної мережі було розширення наших знань про механізми мозку шляхом розробки штучних систем, здатних відтворювати складні (навіть інтелектуальні) обчислення, подібні до тих, які виконує людський мозок. Більшість нейронних мереж використовують правила навчання на основі даних для налаштування вагових коефіцієнтів з’єднань. Іншими словами, нейронні мережі зазвичай розробляються на прикладах. Тоді вони мають певну здатність узагальнювати дані, яких немає в базі навчання.

Таким чином, техніка нейронної мережі є методом регресії, подібним до методів лінійної або багатолінійної регресії. Після того, як параметри (ваги) налаштовано, нейронна мережа є нелінійною статистичною моделлю. Перевага нейронних мереж перед звичайними методами регресії (тобто лінійна регресія, логістична регресія) полягає в тому, що вони, як правило, вимагають більшої кількості настроюваних параметрів для отримання нелінійної моделі заданої точності, зображено на рисунку 2.4.

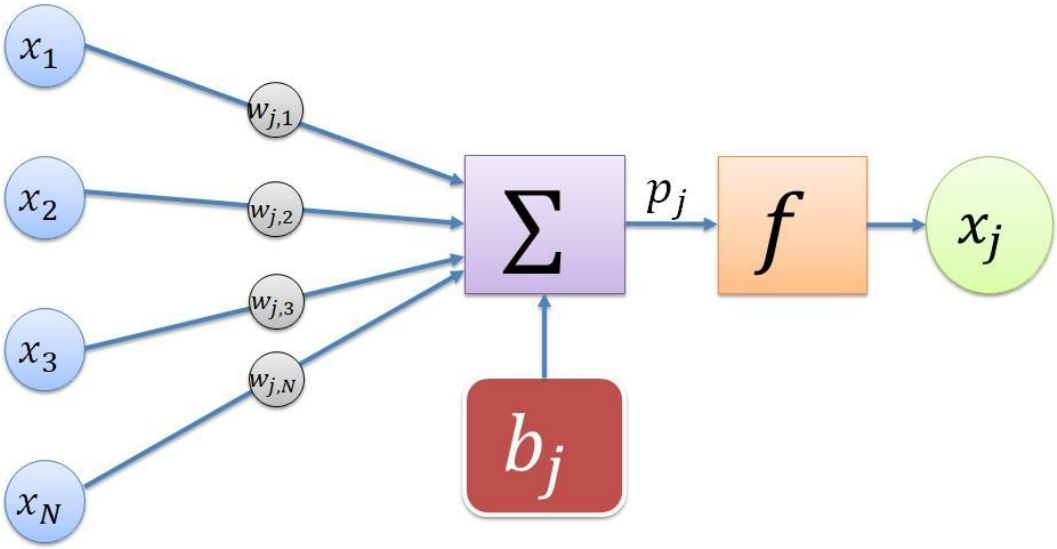


Рис. 2.4. Модель формального нейрона

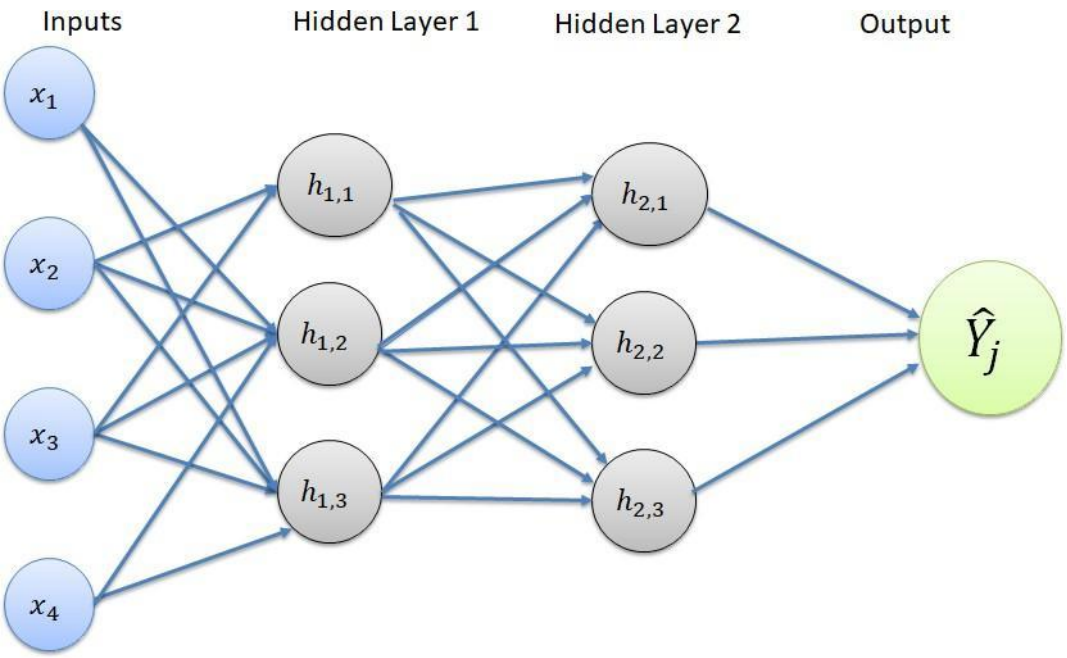
2.4.1. Багатошаровий перцептрон

У 1957 р. Ф.Розенблат був натхненний роботою над формальними нейронами і на правилі Хебба, розробив модель персептрона, а також метод, який дозволяє цій моделі начатись. Хоча ця модель має лише один шар, вона вже дозволяє вирішувати прості задачі класифікації геометричних символів. Однак за допомогою методу, сформульованого Ф.Розенблатом, неможливо навчити систему, що має кілька рівнів, яка через кілька років виявляється дуже обмеженою.

У результаті М. Мінський і С. Пейперт продемонстрували у 1969 році шляхом ретельного аналізу, що персептрон не здатний вивчати функції, якщо вони не є лінійно роздільними. Вони пішли ще далі, продемонструвавши, що для вирішення проблеми необхідно мати принаймні один додатковий шар нейронів. Але на той час не було способу навчити багатошаровий перцептрон. Знадобилося більше десяти років дослідникам для впровадження методу навчання, що дозволяє регулювати параметри багатошарового персептрона. Це був П. Вербос, який першим запропонував ідею використання методу зворотного розповсюдження градієнта помилки, розробленого в 1960-х роках для навчання ШНМ. Під час дипломної роботи, яку він захистив у 1974 р., він проаналізував актуальність цього методу, але, враховуючи відсутність інтересу наукової спільноти до ШНМ після публікації М. Мінського та С. Пеперта, він не опублікував жодних результатів з цього питання до 1982 року.

Нарешті в середині 1980-х років кілька дослідницьких груп заново відкрили метод і остаточно популяризовали. У 1986 р. Д. Румельхарт, Г. Хінтон і Р. Вільямс показали що за допомогою зворотного поширення градієнта помилок, застосованого до багатошарового персептрона, нарешті вдалося подолати обмеження персептрона, які були підняті М. Мінським і С. Вільямсом. Паперт у 1969 році. Зокрема, багатошаровий персептрон може обробляти складні нелінійні проблеми та може апроксимувати, за допомогою одного прихованого шару та достатньої кількості нейронів, будь-яку неперервну нелінійну функцію в компактному просторі з довільною точністю.

Таким чином, багатошаровий персептрон називається апроксиматором універсальної функції. Багатошаровий перцептрон є частиною багатошарових



ациклічних нейронних мереж. Їхні нейрони та їхні зв’язки утворюють ациклічний орієнтований граф, у якому інформація тече лише в одному напрямку (тобто від входу до виходу). Перший шар називається вхідним, проміжний(і) шар(и) називається прихованим шаром, а останній шар називається вихідним. Рисунок 2.5. дає представлення багатошарового персептрона з двома прихованими шарами.

Рис. 2.5. Багатошаровий персептрон із двома прихованими

Активації вхідного рівня отримують інформацію, надану вхідними векторами кожного екземпляра. Тому цей рівень не має жодних вхідних з’єднань з інших вузлів. Однак він повністю пов'язаний з першим прихованим шаром. У MLP, що містить N прихованих шарів, кожен із (N-1) прихованих шарів повністю з’єднаний із верхнім. Прихований шар повністю з’єднаний із вихідним шаром. Активація нейронів у вихідному шарі представляє значення вихідного вектора MLP. Нейрони цих проміжних рівнів, а також нейрони вихідного рівня застосовують дві обробки: лінійну комбінацію їхніх вхідних даних (чиї ваги є параметрами мережі), за якою слідує нелінійна функція (тобто функція активації). Багатошарові персептрони зазвичай використовуються для проблем контрольованої класифікації, але рідше для

обробки даних часових рядів. Це передбачає наявність набору пар входів-виходів, пов'язаних певним відношенням, яке мережа буде «навчатися», регулюючи свої параметри. Навчання виконується за допомогою градієнтного алгоритму зворотного поширення. Проте ці мережі мають деякі недоліки. Насправді фаза навчання може тривати від кількох хвилин до кількох годин залежно від складності проблеми. Крім того, не існує офіційної методології проектування та будівництва такого типу мережі. Вибір гіперпараметрів мережі потребує певного досвіду, щоб отримати бажану продуктивність.

2.4.2. Рекурентна нейронна мережа

В останні роки спостерігалося відновлення використання повторюваних нейронних мереж для обробки часових рядів, особливо в задачах прогнозування. На відміну від раніше розглянутого MLP, мережа рекурентних нейронів — це мережа, граф зв’язків якої містить принаймні один цикл.

Архітектура цього типу мережі призначена для маніпулювання послідовностями вхідних векторів. Вони пам’ятають те, що було обчислено в минулому, що робить їх особливо придатними для обробки послідовностей. Теоретично мережі повторюваних нейронів можуть зберігати в пам’яті інформацію, яку бачать у довільно великій послідовності, але на практиці втрачають свою ефективність у дуже довготривалій динаміці. Саме в цьому сенсі останні роботи показали появу повторюваних архітектур нейронних мереж із механізмами «воріт», які дозволяють значно покращити ємність пам’яті моделей. Дійсно, при кумулятивних обчисленнях протягом тривалого періоду помилка, отримана при зворотному поширенні градієнта, зменшується або, рідше, зростає експоненціально щодо масштабу часу. Ці дві проблеми називаються відповідно «зникаючим градієнтом» і «вибухаючим градієнтом». Ми також зауважимо, що цей тип проблеми також існував у глибоких незациклених архітектурах. Розсіювання або вибух градієнта погіршується в цьому випадку в залежності від кількості шарів. Одне з найефективніших рішень для подолання цієї проблеми обчислення градієнта

проявляється в розширенні концепції RNN, а саме в архітектурі довготривалої короткочасної пам’яті (LSTM).

2.5. Методи тестування програмного забезпечення

Існуючі на сьогоднішній день методи тестування ПЗ не дозволяють однозначно і повністю виявити всі дефекти і встановити коректність функціонування аналізованої програми, тому всі існуючі методи тестування діють у рамках формального процесу перевірки досліджуваного або розроблюваного ПЗ.

Такий процес формальної перевірки, чи верифікації, Може довести, що дефекти відсутні з точки зору використовуваного методу. (Тобто немає ніякої можливості точно встановити або гарантувати відсутність дефектів у програмному продукті з урахуванням людського фактора, присутнього на всіх етапах життєвого циклу ПЗ).

Існує безліч підходів до вирішення завдання тестування і верифікації ПЗ, але ефективне тестування складних програмних продуктів - це процес у вищій мірі творчий, не зводиться до слідування строгим і чітким процедурам або створенню таких .

З точки зору ISO 9126, якість програмного забезпечення можна визначити як сукупну характеристику досліджуваного ПЗ з урахуванням наступних складових:

- надійність;
- супроводжуваність;
- практичність;
- ефективність;
- мобільність;
- функціональність.

Існує кілька ознак, за якими прийнято проводити класифікацію видів тестування. Зазвичай виділяють наступні:

- за об'єктом тестування:
  - функціональне тестування (functional testing);

- тестування продуктивності (performance testing);
- навантажувальне тестування (load testing);
- стрес-тестування (stress testing);
- тестування стабільності (stability / endurance / soaktesting);
- юзабиліті-тестування (usability testing);
- тестування інтерфейсу користувача (UI testing);
- тестуваннябезпеки (security testing);
- тестування локалізації (localization testing);
- тестування сумісності (compatibility testing);

- занаявністю доступу до системи:
  - тестування Чорної Скриньки (blackbox);
  - тестування білої скриньки (whitebox);
  - тестування сірої скриньки (greybox);
- за ступенем автоматизації:
  - ручне тестування (manual testing);
  - автоматизоване тестування (automated testing);
  - напівавтоматизоване тестування (semiautomated testing);
- за ступенем ізольованості компонентів:
  - компонентне (модульне) тестування (component/unittesting);
  - інтеграційне тестування (integration testing);
  - системне тестування (system/end-to-end testing);
- за часом проведення тестування:
  - альфа-тестування (alpha testing);
  - тестування при прийомі (smoke testing);
  - тестування нової функціональності (newfeature testing);
  - регресійне тестування (regression testing);
  - тестуванняприздачі (acceptance testing);
  - бета-тестування (beta testing);
- за ознакою позитивності сценаріїв:

- позитивне тестування (positive testing);
- негативне тестування (negative testing);
- за ступенем підготовленості до тестування:
- тестування по документації (formal testing);
- тестування adhoc або інтуїтивне тестування (adhoc testing).

Необхідність автоматизації тестування класифікується за наступними ознаками:

1) за об’єктом тестування:

– функціональному тестуванні (functional testing);

– тестуванні продуктивності (performance testing);

– навантажувальному тестуванні (load testing);

– стрес-тестуванні (stres stesting);

– тестуваннястабільності (stability / endurance / soaktesting);

2) занаявністю доступу до системи:

– тестування чорної скриньки (blackbox);

3) за ступенем ізольованості компонентів:

– системне тестування (system/end-to-endtesting);

4) за часом проведення тестування:

– альфа-тестування (alpha testing)

– тестування при прийомі (smoke testing)

– регресійнетестування (regression testing)

– тестуванняприздачі (acceptance testing)

– Бета-тестування (beta testing)

Розглянемо більш детально види тестування в яких може застосовуватись автоматизація:

1) функціональне тестування – це тестування ПЗ в цілях перевірки можливості реалізації функціональних вимог, тобто здатності ПЗ в певних умовах вирішувати задачі, необхідні користувачам.

Функціональні вимоги визначають, що саме робить ПЗ, які задачі воно вирішує.

Функціональні вимоги включають в себе:

– функціональну пригодність (англ. suitability);

– точність (англ. accuracy);

– здатність до взаємодії (англ. interoperability).

– Відповідність стандартам і правилам (англ.compliance).

– Захищеність (англ. security).

2) тестування продуктивності в інженерії програмного забезпечення - тестування яке проводиться з метою визначення, як швидко працює система або її частину під певним навантаженням. Також може служити для перевірки і підтвердження інших атрибутів якості системи, таких як масштабованість, надійність і споживання ресурсів.

Тестування продуктивності – це одна зі сфер діяльності розвивається в галузі інформатики інженерії продуктивності, яка прагне враховувати продуктивність на стадії моделювання та проектування системи, перед качаном Основному стадії кодування. У тестуванні продуктивності розрізняють наступні напрямки:

– стрес (stress);

– навантажувальне (load);

– тестування стабільності (endurance or soakorstability);

– конфігураційне (configuration).

3) навантажувальне тестування (англ. Load Testing) – визначення або збір показників продуктивності і годині відгуку програмно-технічної системи або пристрою у відповідь на зовнішній запит з метою встановлення відповідності вимогам, що пред'являються до даної системи (пристрою). Для дослідження годині відгуку системи на високих або пікових навантаженнях проводиться стрес-тестування, при якому створювана на систему навантаження перевищує нормальні сценарії його використання. Не існує чіткої межі між навантажувальним та стрес-тестуванням, однак ці поняття не варто змішувати, так як ці види тестування відповідають на різні бізнес-питання і використовують різну методологію.

Термін тестування навантаження може бути використаний у різних значеннях в професійному середовищі тестування ПЗ. У загальному випадку він означає

практику моделювання очікуваного використання додатка за допомогою емуляції роботи декількох користувачів одночасно. Таким чином, подібне тестування найбільше підходить для екстермінатуса мультикористувацьких систем, частіше - використовують клієнт-серверну архітектуру (наприклад, веб-серверів). Однак і інші типи систем ПЗ можуть бути протестовані подібним способом. Наприклад, текстовий або графічний редактор можна змусити прочитати дуже великий документ; а фінансовий пакет - згенерувати звіт на основі даних за декілька років. Найбільш адекватно спроектований навантажувальний тест дає більш точні результати.

Основна мета навантажувального тестування полягає в тому, щоб, створивши певну очікувану в системі навантаження (наприклад, за допомогою віртуальних користувачів) і, звичайно, використавши ідентичне програмне і апаратне забезпечення, спостерігати за показниками продуктивності системи. В ідеальному випадку в якості критеріїв успішності навантажувального тестування виступають вимоги до продуктивності системи, які формулюються і документуються на стадії розробки функціональних вимог до системи до початку програмування основних архітектурних рішень. Однак часто буває так, що такі вимоги не були чітко сформульовані або не були сформульовані зовсім. У цьому випадку перше навантажувальний тестування буде являтися пробним (exploratoryloadtesting) і ґрунтуватися на розумних припущеннях про очікувану навантаженні і споживанні апаратної частини ресурсів.

Розробка методів побудови математичних моделей "чорної" скриньки є однією з важливих кібернетичних проблем. За умови наявності математичної моделі "чорної" скриньки з'являється можливість віднести його до якогось одного класу, всі системи якого ізоморфні по поведінці.

Для науки метод “чорної скриньки” має дуже велике значення. З його допомога в науці було зроблено дуже багато видатних відкриттів. Наприклад, вчений Гарвей Галі в XVII столітті передбачив будову серця. Він моделював роботу серця насосом, запозичивши ідеї із зовсім іншої області сучасних йому знань - гідравліки. Практична цінність методу “чорної скриньки” полягає по-перше, в

можливості дослідження дуже складних динамічних систем, і, по-друге, в можливості заміни одного "ящика" іншим. Навколишня дійсність і біологія дають масу прикладів виявлення будови систем методом "чорної" скриньки.

Висновок

Зважаючи на описані у першому розділі обставини з розробки програмного засобу системи розумного будинку з використанням методів обліку енергоефективності, а саме управління витратами будинку, автоматизованого аналізу та розрахування дій для комфортного використання користувачем.

Аналіз існуючих підходів енергомоніторингу, методів машиного навчання показав, що при розробці систем доцільно використовувати перевірені методи проектування, уважно вивчати рекомендації до розробки систем і проводити тестування на тестових стендах.

Було визначено, що програмний засіб обліку енергоефективності в системі «Розумний будинок» є обов’язковим елементом в теперішній час. Використання енергомоніторингу та технологій машиного навчання необхідна для звичайних користувачів, як обов’язковий елемент швидкого обчислення та прийняття рішень при зміні графіку перебування людей в будинку.

Для забезпечення якості використання програмного засобу, необхідно проводити тестування застосунку на великій кількості клієнтів.

Обравши методи енегромоніторингу та методи машиного начання, я переходжу до аналізу чітких та статичних вимог для створення такого застосунку.

РОЗДІЛ 3
ТЕХНОЛОГІЇ ТА КОМПОНЕНТИ ПРОГРАМНОГО ЗАСОБУ

3.1. Ідея програмного засобу «Розумний будинок»

У цьому розділі буде надано інформацію про систему, побудовану в дипломній роботі, а також представлено різні компоненти та технології, які використовуються для реалізації. У цьому розділі будуть розглянуті різні платформи домашньої автоматизації. Далі буде інформація про те, які інструменти та обладнання будуть використовуватися для системи відстеження користувачів. Будуть представлені розумні домашні пристрої, які будуть використовуватися в цій дипломній роботі.

Створення центральної системи розумного будинку на основі IoT, яка могла б контролювати та автоматизувати різні пристрої розумного дому в будинку. Ця система використовуватиме невеликий одноплатний комп'ютер і можливості платформи з відкритим вихідним кодом, які відрізняються гарною модульністю, простотою розробки та низькою вартістю. Система відстеження користувачів буде створена з метою збору даних про місцезнаходження користувача в будинку. Ці дані використовуватимуться для автоматизації інтелектуальних пристроїв у домі, щоб вони реагували, коли користувачі знаходяться поруч із ними, або покинули їх зону дії. Основна мета взаємодії між відстеженням місцезнаходження користувача та керуванням розумним пристроєм — показати, на що здатна система розумного будинку на основі IoT.

Відстежується не лише місцезнаходження користувачів, а й пристрої розумного будинку. Система «розумного дому» використовуватиме дані, зібрані з кожного пристрою «розумного дому», і забезпечить ручну панель керування для доступу користувачів будинку. Зазначена панель керування відображатиме стан кожного пристрою, матиме можливість керувати цими пристроями вручну та, можливо, відображатиме будь-які корисні для користувача дані, наприклад, час, дату, клімат у домі та поза ним, споживання енергії тощо. Ця панель керування буде працювати як у формі веб-додатку.

Приклад можливого сценарію, який має бути можливим із цією системою: Користувач приходить додому. Система виявляє це і вмикає світло в тамбурі та гардеробі. Далі, оскільки зараз теплий літній день, система увімкне кондиціонер. Коли Джо досягне своєї кімнати, система вимкне світло в попередніх місцях і ввімкне світло в кімнаті користувача.

Через деякий час користувач йде, і система вимикає всі активні пристрої, і оскільки температура в приміщенні все ще відносно висока, кондиціонер налаштований на роботу в еко-режимі для більш ефективного споживання енергії.

Пізніше інший користувач приходить додому і починає прати білизну. Коли людина працює в пральні та ходить по дому, збираючи старі простирадла, система автоматично вмикає та вимикає світло в кімнатах, які вона відвідує.

Згідно з цим прикладом, система повинна працювати з кількома користувачами та мати можливість розрізняти їх. Це також уможливить індивідуальну автоматизацію. У наступних підрозділах буде представлено компоненти та інструменти, які використовуються для створення цієї системи.

3.2. Платформа домашньої автоматизації

Платформа домашньої автоматизації — це система, яка відповідає за моніторинг та керування домашніми пристроями, такими як освітлення, клімат, розважальні системи, побутова техніка та навіть системи безпеки. Ці пристрої зазвичай підключаються до центрального концентратора, який керується платформою домашньої автоматизації. Є багато конкуруючих постачальників для цих платформ, і навіть кілька систем з відкритим кодом.

Для цієї дипломної роботи платформа домашньої автоматизації повинна мати можливість ефективно обробляти кілька пристроїв і мати підтримку простого й ефективного зв’язку між пристроями, а також підтримку великої кількості пристроїв і пристроїв розумного дому. Ці платформи також повинні мати добре підготовлену документацію для розуміння можливостей і обмежень платформ, а також мати активну базу користувачів і спільноту для швидкого отримання відповідей на будь-яке конкретне запитання.

Ще одним критерієм для платформи домашньої автоматизації є підтримка Інтернету речей і ефективне використання ресурсів. Оскільки ця система розумного дому використовуватиме невеликий одноплатний комп'ютер, бажану платформу потрібно буде оптимізувати для роботи на ньому. Стабільність також є важливим фактором, і не повинно виникнути жодних проблем із збереженням і безпекою зібраних даних протягом певного часу.

У наступних підрозділах буде представлено кілька платформ, які потенційно задовольняють встановлені критерії та можуть бути використані для створення подібних систем, створених у цій дипломній роботі. На основі цих платформ буде вибрано та представлено та обговорено платформу, використану в цій дипломній роботі. Також представлена одна комерційна платформа для кращого порівняння.

3.2.1 Платформа Home Assistant

Програмне забезпечення для домашньої автоматизації Home Assistant - це безкоштовна платформа з відкритим вихідним кодом, для створення центральних систем керування пристроями та технікою розумного будинку. Ця платформа створена й написана на Python. З листопада 2020 року, згідно з офіційною сторінкою інтеграції, Home Assistant має понад 1700 модульних доповнень, які дозволять використовувати різноманітні пристрої, сервіси та системи розумного дому.

Проект Home Assistant стартував у вересні 2013 року, а через кілька місяців у листопаді основний проект був уперше публічно опублікований на GitHub. Засновником цього проекту є Паулюс Шоутсен, і, за його словами, метою цього проекту було стати платформою для будинку. З моменту першого випуску проект був скоріше проектом хобі, спрямованим на керування світлом Philips Hue. Зараз кількість спільноти Home Assistant зросла, і в 2020 році GitHub поставив Home Assistant на друге місце серед пакетів Python із найбільшою кількістю учасників.

Цей проект мав на меті створити платформу розумного дому, яка могла б контролювати все з централізованої точки. Його ідея полягала в тому, щоб створити платформу розумного будинку, яка зосереджена на зручності та адаптивності систем розумного будинку. Система домашньої автоматизації не повинна бути чимось, що

стане громіздким у використанні, а натомість бути тим, що всі члени будинку знайдуть корисним. Це означає, що система повинна працювати бездоганно, поєднується з повсякденним робочим процесом і має працювати вдома. Загалом, згідно з дописом у блозі, бачення Home Automation полягало в тому, щоб мати можливість розробити системи розумного дому, які ніколи не будуть заважати чи дратувати, але їх пропускатимуть, коли вони не працюватимуть.

Home Assistant можна використовувати різними способами. Рекомендованим способом було б використовувати їхню систему в спеціальній системі, наприклад, на Raspberry Pi. Він також доступний як рішення на основі контейнерів, яке, наприклад, може працювати на Docker. Після початкової конфігурації платформа вже могла виявляти деякі розумні побутові прилади в домашній мережі. Інші пристрої, які не були знайдені автоматично, можна легко додати через інтерфейс користувача. якщо для цих пристроїв доступна інтеграція.

Відповідно до сторінки документації Home Assistant, автоматизацію в можна здійснити різними способами. Найпростіший спосіб реалізувати автоматизацію через інтерфейс користувача за допомогою функції редактора автоматизації. Це створює автоматизацію, яка працює, чекаючи на тригер, потім перевіряючи умови та, нарешті, викликаючи дію. Для більш просунутої автоматизації Home Assistant має функцію шаблонів, яка використовує систему шаблонів Jinja2 і синтаксис. Використовуючи це, користувачі можуть отримати вигоду від більшої різноманітності операцій і користувальницьких змінних для побудови більш глибокої автоматизації та системних правил. Домашній помічник також може запускати сценарії python. Це означає, що користувачі не обмежуються використанням лише наданого механізму створення шаблонів, а можуть реалізувати свої функції, які обмежені лише можливостями Python та апаратним забезпеченням.

3.2.2. Apple HomeKit

HomeKit Згідно зі сторінкою розробника Apple, це основа для пристроїв і приладів розумного дому, для яких виробники підтримують HomeKit[24]. Ця програмна в основному працює лише з програмами, які працюють на операційній

системі Apple. Такі додатки, дозволяють користувачам налаштовувати, контролювати та автоматизувати свої розумні пристрої.

Apple HomeKit вперше було випущено з iOS 8 у 2014 році[16]. Під час випуску фреймворк працював зі сторонніми додатками, що дозволило їм мати інтерфейс для пристроїв HomeKit. У 2016 році Apple випустила власну офіційну програму під назвою Home для керування всіма пристроями з підтримкою HomeKit і завдяки цьому стала можливою проста автоматизація. Програма працювала лише з пристроями під керуванням iOS 10 і watchOS 3, але пізніше, у 2018 році, ця програма також була випущена на комп'ютерах Apple під керуванням macOS 10.14.

За словами Apple, головною ідеєю HomeKit було спрощення завдань у домі. Для цього HomeKit був розроблений, як набір інструментів, що дозволяє Apple та іншим людям створювати рішення для своїх потреб. Ці рішення можуть забезпечити високий рівень керування аксесуарами та пристроями.

Незважаючи на те, що користувачі можуть використовувати HomeKit для розробки власного комплексного рішення для розумного дому, для моніторингу, керування або автоматизації пристроїв із підтримкою HomeKit, вона має мінус перед платформами з відкритим кодом. Цей мінус полягає в тому, що кількість підтримуваних пристроїв невелика. Станом на 2019 рік лише 450 пристроїв були позначені як сумісні. Це може бути пов'язано з програмою MFI, до якої виробники повинні зареєструватися, щоб отримати дозвіл на додавання можливостей HomeKit до своїх пристроїв. Це означає, що лише певні виробники можуть розробляти пристрої, які працюють з HomeKit.

3.3. Система відстеження користувачів

Ця дипломна робота має на меті показати, чого IoT може досягти в середовищі розумного будинку. У цій дипломній роботі буде розроблено систему відстеження користувачів, яка могла б збирати дані про місцезнаходження користувача вдома та передавати ці дані через MQTT на платформу домашньої автоматизації. Система

відстеження користувачів, по суті, забезпечить більш складну автоматизацію та керування системою розумного дому завдяки можливостям IoT[39].

Системи відстеження користувачів можуть бути реалізовані в різних формах. Одним зі способів їх розділення була б система, яка могла б або не могла розрізняти користувачів. Обидві системи мають свої варіанти використання, методологію та реалізації. Для системи, яка не може розпізнати одного користувача від іншого, це зазвичай реалізується для визначення руху або існування сутності навколо системи. Для цього ці системи використовуватимуть датчики, які можуть виявляти рух або випромінювання тепла від сторонньої організації. Прикладом того, де ці системи найчастіше зустрічаються, є система безпеки для виявлення вторгнень.

Системи відстеження, які можуть розрізняти користувачів у діапазоні виявлення, зазвичай реалізуються для відстеження місцезнаходження користувачів. Це робиться з кількох причин. Одна з них — для безпеки. Якщо можна знати, де хтось знаходиться, то можна закрити доступ до зон, до яких вони не мають прав. Багато офісів використовують цей підхід, щоб утримувати персонал у визначених місцях.

Другий варіант використання — це контроль і автоматизація середовища навколо користувача. Це в основному реалізовано в рішеннях розумного будинку для більш складної, персоналізованої та зручної автоматизації. Прикладом системи домашньої автоматизації з цією системою відстеження може бути, коли один користувач може увійти у ванну кімнату, а ванна буде попередньо нагріта до бажаної температури.

Щоб ця система могла розрізняти користувачів і відстежувати їх місцезнаходження, система повинна буде зчитувати ідентифікаційні дані користувача, коли він входить у нову зону. Дані, які використовуються для ідентифікації, можуть бути завжди у користувача, наприклад, біометричні дані або зовнішній носій інформації. У разі використання біометрії система могла б реалізувати розпізнавання обличчя за допомогою камер або сканерів відбитків пальців на кожному вході.

Використовуючи зовнішні носії ідентифікаційних даних, система може бути реалізована різними способами залежно від носія. Носії можуть обмінюватися даними двома способами: контактним або близьким. Для діапазону контактів носії неактивні й лише пасивно обмінюються даними, коли вони контактують із якимось зчитувачем. Ця система може бути реалізована шляхом використання персоналізованих карт-ключів і зчитувачів карток на кожному вході. Що стосується носіїв ближнього радіусу дії, вони можуть бути активними та підключатися до найближчих сканерів для передачі ідентифікаційних даних із діапазону. Це означає, що користувачам не потрібно було б діяти біля кожного входу, а просто проходити повз, і система сама фіксувала б їхнє місцезнаходження.

Для цієї дипломної роботи система відстеження зможе розрізняти користувачів за допомогою зовнішніх носіїв ідентифікаційних даних, які система може зчитувати з близької відстані. Це дозволить безперебійно впроваджувати автоматизацію біля користувачів і буде більш зручним, оскільки користувачеві не потрібно виконувати ідентифікаційну дію на кожному вході. Система складатиметься з кількох трекерів, по одному на кімнату, щоб автоматично виявляти користувача, коли він знаходиться поблизу або в кімнаті.

3.4. Апаратне забезпечення трекерів

Система відстеження користувачів цієї дипломної роботи буде серією автономних пристроїв, які підтримують такі бездротові технології, як WIFI і Bluetooth. Ці пристрої будуть розподілені по будинку, по одному пристрою на кімнату. Для надсилання даних з кожного пристрою на платформу домашньої автоматизації знадобиться WIFI.

Бездротові технології також будуть використовуватися для відстеження користувачів. Щоб виявити користувачів, пристрої відстеження в різних кімнатах використовуватимуть Bluetooth для пошуку пристроїв Bluetooth, які носять користувачі. Таким чином, якщо буде виявлено пристрій Bluetooth користувача, наприклад його телефон, трекер може перевірити силу сигналу від нього до

пристрою. Після однієї ітерації сканування пристрій надішле дані про виявлені пристрої та рівень їх сигналу через MQTT[37].

Щоб реалізувати цю ідею, автономні пристрої повинні бути програмованими та здатними працювати безперервно. Можна розглянути кілька різних апаратних варіантів. Двома основними варіантами є використання одноплатних мікроконтролерів Arduino або одноплатних комп'ютерів Raspberry Pi. Обидва варіанти мають свої сильні та слабкі сторони.

Arduino — це компанія з відкритим вихідним кодом, яка розробляє та виробляє одноплатні мікроконтролери, відповідно до вступної сторінки Arduinos. Серед широкого спектру контролерів, які вони виробляють, деякі можуть використовувати WIFI і Bluetooth. По суті, мікроконтролери Arduino ідеально підходять для взаємодії з датчиками та обробки даних у невеликих кількостях. Залежно від варіанту використання вони можуть бути дуже доступними. Основним недоліком мікроконтролерів Arduino є те, що вони здатні виконувати переважно невеликі одиничні завдання, а деякі контролери нелегко знайти.

Raspberry Pis розроблено Raspberry Pi Foundation у Сполученому Королівстві. Raspberry Pi — це одноплатний комп'ютер, який може запускати операційні системи та графічний вивід. Сфера їх використання ширша порівняно з мікроконтролерами Arduino, оскільки вони не обмежені апаратним забезпеченням і доступні за ціною.

Окремі пристрої системи відстеження в цій дипломній роботі використовують одноплатні комп'ютери Raspberry Pi Zero W. Хоча Arduino також має мікроконтролери, які підходять для використання системи відстеження, Raspberry Pi Zero Ws був більш доступним. Крім того, ці Raspberry Pis дозволили більше майструвати та досліджувати різні ідеї виявлення користувачів.

3.5. Програмне забезпечення трекерів

Щоб трекери працювали, їм потрібно було мати програмне забезпечення, здатне використовувати апаратне забезпечення. Оскільки трекери працюють під управлінням операційної системи Raspberry Pi OS Lite на базі Linux, існувало кілька

різних способів реалізації відстеження та виявлення користувачів. Одним із способів було використання знайомої мови програмування та її модулів для реалізації виявлення Bluetooth і передачі даних через MQTT. Іншим варіантом було знайти вже існуюче рішення з відкритим вихідним кодом і розробити на його основі програмне забезпечення.

Після пошуку можливих рішень з відкритим вихідним кодом було знайдено лише одне, яке підходило для використання під назвою reelyActive. reelyActive було засновано в 2012 році з ідеєю створити хмарну активну систему RFID. З роками їхня команда та проект зростали, і станом на 2017 рік рішення може працювати на Raspberry Pi та підтримує ідентифікацію пристроїв Bluetooth.

ReelyActive здається чудовим рішенням з відкритим вихідним кодом, яке можна використовувати для реалізації програмного забезпечення для трекерів. Але через застарілу та майже неіснуючу документацію та підтримку, здавалося б, лише технології Bluetooth Low Energy, це рішення з відкритим кодом не використовувалося. Оскільки відповідних рішень із відкритим вихідним кодом для реалізації програмного забезпечення трекерів не було, єдиним способом було впровадження рішення з нуля. Програмне забезпечення системи відстеження користувачів буде реалізовано на Python, оскільки воно підтримує Bluetooth і MQTT[25].

Висновок

У цьому розділі було надано інформацію про систему, побудовану в дипломній роботі, а також представлено різні компоненти та технології, які використовуються для реалізації програмного засобу розумного будинку.

Була представлена інформація про те, які інструменти та обладнання будуть використовуватися для системи відстеження користувачів. Були представлені розумні домашні пристрої, які будуть використовуватися.

РОЗДІЛ 4
РЕАЛІЗАЦІЯ ПРОТОТИПУ ПРОГРАМНОГО ЗАСОБУ

4.1. Функціональні вимоги

У цьому підрозділі буде описано функції, які розумний дім і система відстеження нададуть користувачам. Опис слідуватиме за схемою функціональності, а потім – за описом функціональності.

- Система розумного дому повинна показувати дані про підключені пристрої. Це означає, що користувачі повинні мати можливість бачити будь-яку значущу інформацію про пристрої, підключені до системи розумного дому, за допомогою інтерфейсу користувача. Наприклад, дані можуть стосуватися стану живлення пристроїв або інформації, зібраної датчиками.

- Користувач повинен мати можливість вручну керувати пристроями. Пристрої, які підключені до системи «розумний дім» і якими можна керувати, повинні бути доступними для користувача. Це означає, що користувач повинен мати можливість використовувати інтерфейс користувача для керування пристроями будь-яким значущим способом.

- Користувач повинен мати можливість легко додавати нові пристрої до системи розумного дому. Це означає, що система розумного дому повинна мати функціональні можливості для легкого додавання будь-якого підтримуваного пристрою. Наприклад, ця система може знаходити нові пристрої, переглядаючи пристрої, підключені до мережі.

- Система розумного будинку повинна дозволяти писати сценарії та правила автоматизації. Це означає, що програмний засіб «розумний дім», яка використовується в цій дипломної роботи, повинна дозволяти писати власні сценарії автоматизації для підключених до неї пристроїв.

- Система розумного будинку повинна мати можливість зберігати дані в ній. Це означає, що система «розумний дім» повинна мати можливість автоматично або вручну зберігати інформацію в ній або в хмарі. Це призначено для збереження станів пристроїв для цілей автоматизації та моніторингу та створення графіків із

використанням вибраних даних, зібраних за певний час.

● Розташування гостей має бути включено до системи автоматизації на основі розташування в системі розумного будинку. Це означає, що система розумного дому може використовувати інформацію про місцезнаходження гостей і забезпечувати загальну автоматизацію на основі їхнього місцезнаходження.

● Система відстеження користувачів повинна мати можливість часто обмінюватися інформацією з системою розумного будинку. Це означає, що система розумного будинку повинна мати доступ до інформації про місцезнаходження користувача, яка часто оновлюється. Це дає змогу використовувати автоматизацію на основі місцезнаходження користувача, яка може запускатися щоразу, коли користувач знаходиться поблизу певної кімнати чи місця.

● Трекери системи відстеження користувачів слід підключати та вимикати без проблем. Це означає, що користувачі можуть відключити трекери від електрики та знову підключити їх без жодних проблем. Після ввімкнення трекери повинні автоматично розпочати пошук користувачів.

● Система відстеження користувачів повинна мати можливість відстежувати місцезнаходження гостей. Це означає, що трекери системи відстеження користувачів відстежуватимуть не лише користувачів, але й будь-якого невідомого користувача та їхні пристрої. Це дозволить системі автоматизації розумного дому також працювати з гостьовими місцями.

4.2. Нефункціональні вимоги

У цьому підрозділі будуть описані нефункціональні вимоги, які є критерієм як для системи розумного будинку, так і для системи відстеження користувачів. Описи слідуватимуть за схемою нефункціональності, а потім описом нефункціональності.

● Автоматизація визначення місця розташування користувача системи розумного дому повинна спрацьовувати майже миттєво, коли місцезнаходження користувача змінюється. Це означає, що не повинно бути великої різниці в часі між користувачем, який змінює своє місцезнаходження, і активацією автоматизації на основі місцезнаходження.

- Система «розумного дому» повинна дозволяти одночасному запуску кількох автоматичних систем. Це означає, що система може працювати з кількома автоматизаціями одночасно, що запобігатиме накопиченню завдань. системи «Розумний дім».

4.3 Огляд системи

Система домашньої автоматизації, побудована в цій дипломній роботі, складається з двох підсистем: платформа домашньої автоматизаціїісистема відстеження користувачів.

Платформа домашньої автоматизації буде центральною системою, яка обслуговує інтелектуальні пристрої та послуги, забезпечує автоматизацію та взаємодію з користувачем.

Система відстеження користувачів складається з автономних пристроїв відстеження, кожен з яких відповідає за певні кімнати в будинку. Цю систему відстеження можна розглядати як послугу для платформи домашньої автоматизації. По суті, система відстеження відповідає за збір даних про місцезнаходження користувача та надсилання їх на платформу домашньої автоматизації для автоматизації на основі місцезнаходження користувача. Ці дві підсистеми взаємодіють через мережу MQTT. Мережа складатиметься з брокера MQTT, розміщеного на платформі домашньої автоматизації, і клієнтів MQTT на кожному трекері. Кожен клієнт MQTT на трекерах публікуватиме зібрані дані брокеру MQTT. Платформа домашньої автоматизації також матиме клієнт MQTT, який підписаний на брокера. Це дозволяє платформі керувати надісланими даними та використовувати цю інформацію в автоматизації. Описана комунікаційна мережа системи представлена на рисунку 4.1.

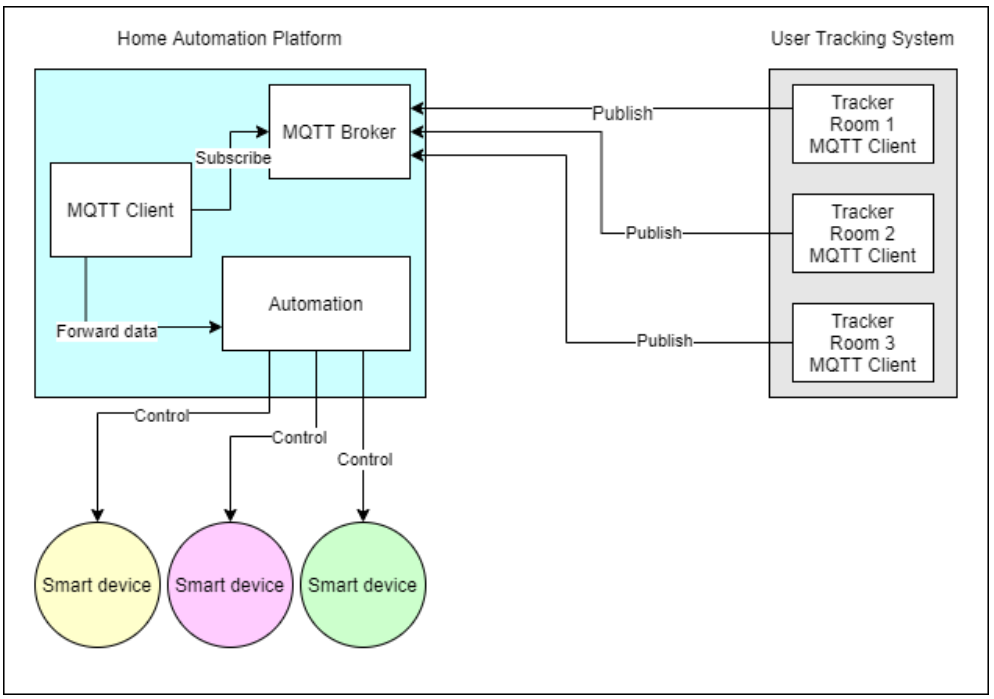


Рис. 4.1. Мережа зв'язку між системами. Платформа домашньої автоматизації

Результатом виконання кваліфікаційної роботи повинна бути розроблений програмний засіб з її функціональними елементами, рисунок 4.2.:

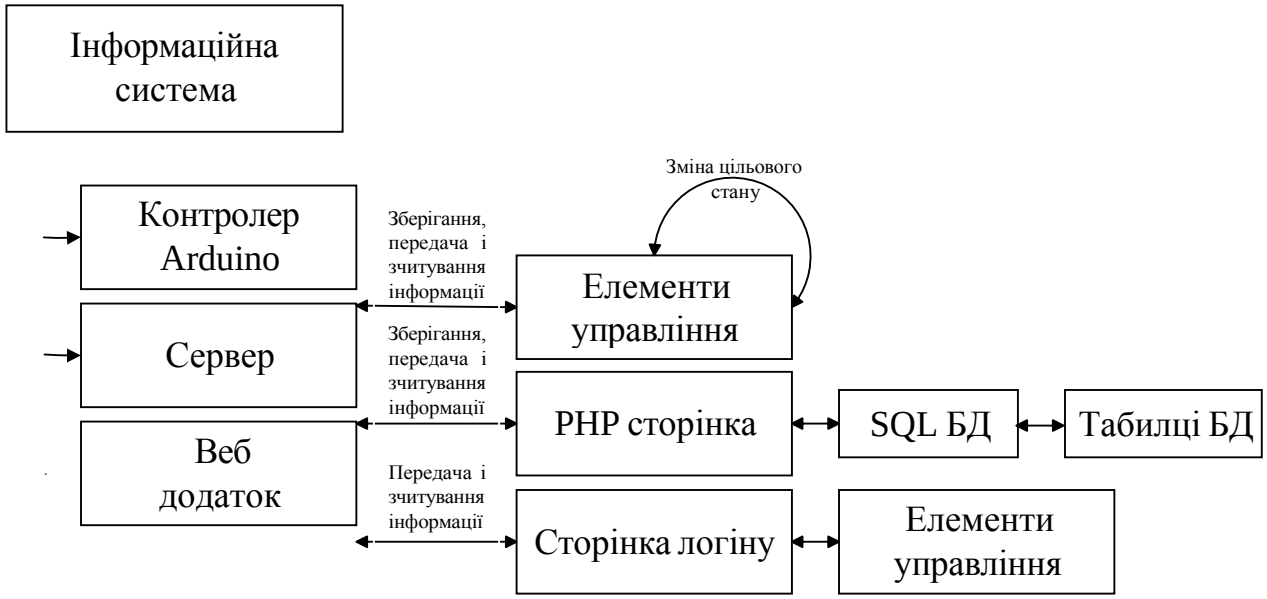


Рис. 4.2. Спрощена схема кінценвого прокукту

Елементи програмного засобу будуть комунікуватися за допомогою мережі інтернет між собою. Причому зв’язок між всіма її елементами проходить через

сервер, в якому за допомогою бази даних і зберігається вся інформація щодо стану елементів розумного будинку, рисунок 4.3.

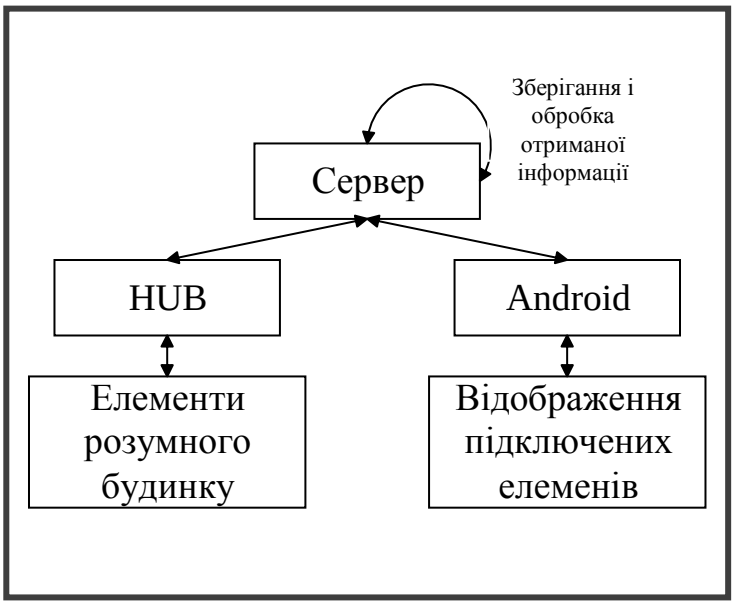


Рис 4.3. Типова схема розумного будинку з концентраційним HUB

Ця схожість основного функціоналу серед різних систем дозволяє розділити систему на три складові, кожна складова підтримує роботу системи в штатному режимі. Перша з них це контролер (HUB), він дозволяє підключити всі елементи будинку до програмного засобу і подальшого управління елементами, також контролер зв’язується з сервером для віддаленого контролю підключеними елементами. Друга з складових це сервер, він оброблює отриману інформацію і зберігає її до бази даних в зручному для перегляду іншими елементами програмного засобу. Сервер забезпечує безперервний доступ до даних. І третій елемент системи це веб додаток. Він дає змогу переглядати і взаємодіяти з елементами будинку в зручній для цього формі. Симбіоз цих елементів дає змогу поліпшити умови проживання в оселі.

Щоб розібрати систему по складовим потрібно дізнатись з яких елементів складається програмний засіб.

Розроблений програмний засіб складеться з:

- контролер на базі мікропроцесора Arduino;
- сервер;

- мобільний додаток на базі ОС Android 7.0 або вище;

Розглянувши спрощену схему UML діаграми класів та UML діаграму послідовності (рисунок 4.4. та рисунок 4.5. розберемо інформаційну систему по компонентам.

Рис. 4.4. UML діаграма класів програмного засобу

На рисунку 4.6. зображена повна схема розташувань виводів мікросхеми Arduino. Всього виводів у мікросхеми 53шт. Звісно для демонстраційної моделі не були використані всі виводи, але якщо реалізовувати інформаційну систему в повному обсязі для проетку великого будинку то їх не буде достатньо в повному обсязі, для цього потрібно буде підключити ще один мікроконтролер через SerialPort, під’єднати вихід на вхід (на рисунку 4.6. зображені як RX0 вхід і TX0 вихід) аналогічно і для другого контролера.

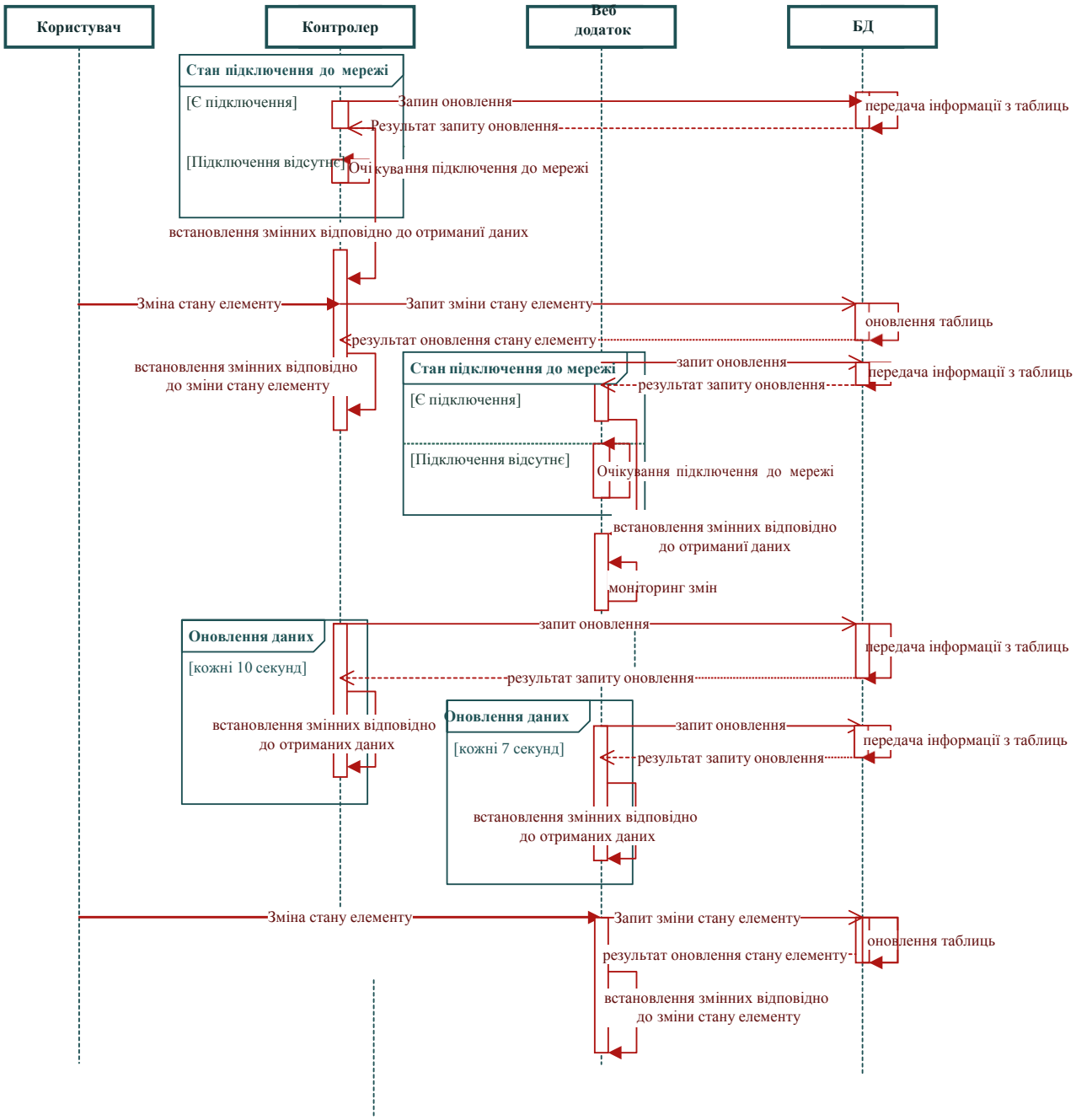


Рис. 4.5. UML діаграма послідовностей програмного засобу

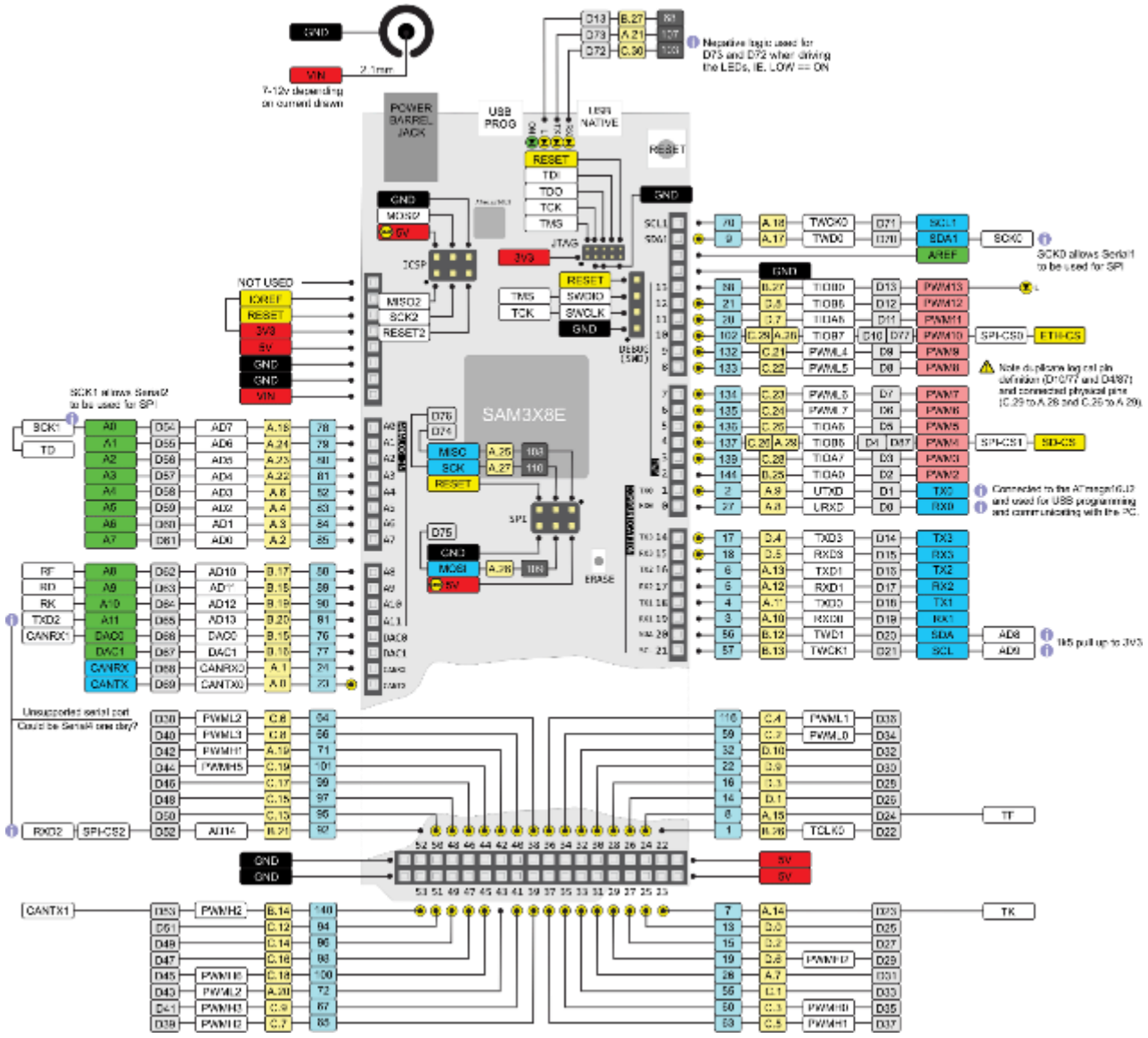


Рис. 4.6. Повна схема розташувань виводів, пінів мікросхеми Arduino

4.4. Система програмного засобу

Платформа домашньої автоматизації складатиметься з двох частин: програмної та апаратної. Програмне забезпечення встановлює вимоги до обладнання. Програмний засіб «Розумний будинок», підходить для роботи на різних системах і обладнанні. Одним із апаратних засобів, які підтримує програмний засіб «Розумний будинок», є лінійка одноплатних комп’ютерів Raspberry Pi.

У цій дипломній роботі використовується Raspberry Pi 4 для розміщення та запуску платформи «Розумний будинок». Raspberry Pi 4 має різні доступні

конфігурації на ринку, і одна, яка використовується в дипломній роботі, має наступні апаратні характеристики:

- ЦП – Broadcom BCM2711, чотирьохядерний Cortex-A72, 64-розрядний, 1,5 ГГц
- Оперативна пам’ять – 4 ГБ LPDDR4-3200 SDRAM
- WIFI – 2,4 ГГц і 5,0 ГГц IEEE 802.11ac
- Bluetooth – Bluetooth 5.0 і BLE

Цей одноплатний комп’ютер не має вбудованої пам’яті, і, таким чином, пам’ять обробляється через карту Micro-SD, на платі є доступний слот для неї. На платі є й інші функції, такі як порти графічного виводу через micro-HDMI, порт Gigabit Ethernet і порти USB 3.0 і 2.0.

Одноплатний комп’ютер Raspberry Pi 4 працюватиме під керуванням системи Linux, налаштованої для програмного засобу «Розумний будинок». Програмний засіб «Розумний будинок» базується на стандартній ОС Raspberry Pi OS Lite. Він має багато функцій, які спрощують налаштування.

Користувач також може налаштувати параметри системи для створення резервних копій програмного засобу «Розумний будинок».

4.5. Початкове налаштування

Налаштування платформи домашньої автоматизації на одноплатному комп’ютері Raspberry Pi вимагає наступного:

- Raspberry Pi 4
- Інструмент Flash
- Карта Micro-SD на 16 ГБ
- Пристрій для читання карт Micro-SD
- Доступ через Ethernet

Програмний код можна записати на картку Micro-SD. Існує багато інструментів, доступних для запису даних, але в цій дипломній роботі використовується balenaEtcher від balena. Цей інструмент відносно простий у

використанні, оскільки він потребує лише кількох вводів від користувача. За допомогою цього інструменту спочатку вибирається файл коду, а потім місце призначення.

Ще одна вимога для запису коду на картку Micro-SD — мати карту Micro-SD із 16 ГБ пам’яті та можливість комп’ютера керувати процесом прошивки. 16 ГБ пам’яті не є обов’язковим для карти Micro-SD, оскільки зображення не займає багато місця, але рекомендовано документацією. Щоб комп’ютер міг отримати доступ до картки Micro-SD, він повинен мати пристрій для читання карток SD, сумісний із картками Micro-SD.

Після завершення процесу перепрошивання наступним кроком буде налаштування програмного засобу «Розумний будинок» на Raspberry Pi

На Raspberry Pi 4 потрібно встановити карту Micro-SD, підключеним до Інтернету через WIFI або Ethernet-кабель. Користуватися кабелем Ethernet легше, оскільки підключення через WIFI передбачає зміну файлу конфігурації з WIFI SSID і паролем перед першим завантаженням. Після встановлення картки Micro-SD і підключення до Інтернету одноплатний комп’ютер можна завантажувати.

Після першого завантаження Raspberry Pi 4 автоматично налаштується. Ця тривалість процесу повністю залежить від підключення до Інтернету, швидкості запису та зчитування з карти Micro-SD і можливостей обробки комп’ютерів Raspberry Pi.

Після завершення процесу буде налаштовано веб-сервер програмного засобу «Розумний будинок». Доступ до інтерфейсу користувача веб-сервера можна отримати з будь-якого комп’ютера в мережі, через веб-браузер із призначеною IP-адресою мережевого маршрутизатора для Raspberry Pi 4 через порт 8080. Наприклад, коли маршрутизатор встановлює IP-адресу для пристрою Raspberry як 192.168.1.3 , тоді до інтерфейсу користувача можна отримати доступ у веб-браузері за адресою 192.168.1.3:8080.

4.6. Конфігурація

Окрім цього, конфігурація може бути спрямована на додавання нових пристроїв, послуг, пакетів, впровадження інтерфейсів користувача та додавання автоматизації. Це можна зробити двома різними способами. Перший варіант полягає у використанні веб-інтерфейсу користувача. За допомогою інструменту Paper UI користувачі могли встановлювати пакети та модулі для додавання різних пристроїв і служб і налаштовувати їх. Цей інструмент також дозволяє відстежувати та контролювати додані пристрої та служби та додавати до них автоматизацію як показано на рисунку 4.7. та рисунку 4.8.

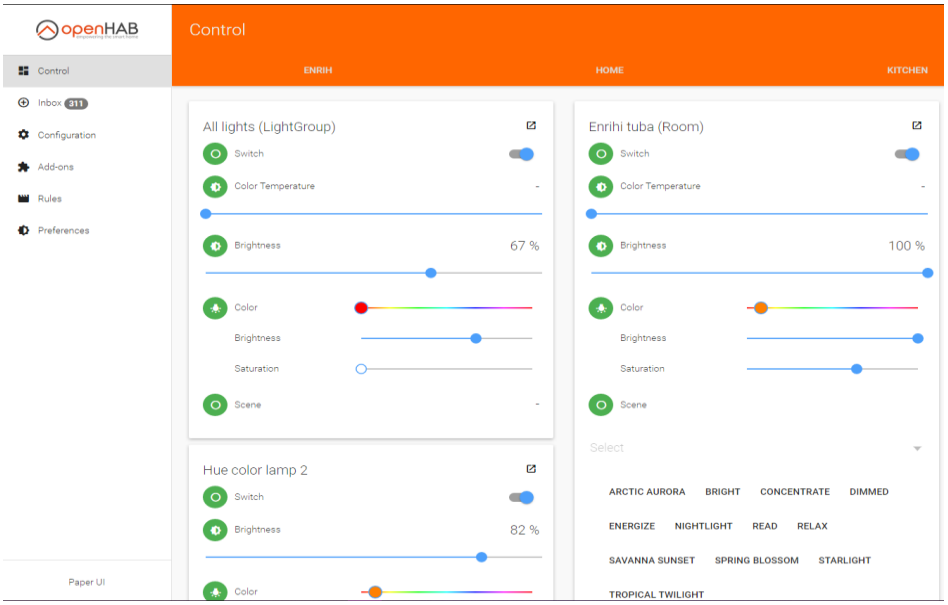


Рис. 4.7. Список інструментів

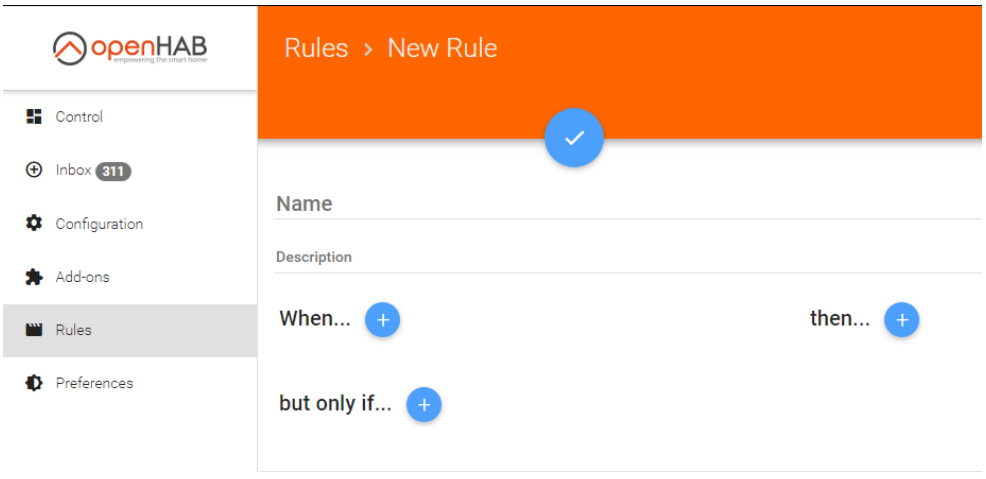


Рис. 4.8. Вид меню керування

Другий варіант — змінити файли конфігурації безпосередньо в системі. Завдяки цій опції користувачі могли досягти тих самих результатів, що й інструмент Paper UI у веб-інтерфейсі. Цей варіант також дозволяє користувачам писати більш просунуту автоматизацію на основі JavaScript, що робить його більш гнучким, ніж перший варіант.

4.6.1.Налаштування програми

Однією з розширених функцій є можливість MQTT, яка за замовчуванням вимкнена. Оскільки система, побудована в цій дипломній роботі, потребує цієї функції, її слід увімкнути.

Для доступу до системи було встановлено з'єднання SSH за допомогою PuTTY. Щоб відкрити сеанс PuTTY у системі, IP-адреса Raspberry Pi 4 і порт SSH потрібні як адресат. IP-адресу було знайдено за допомогою інтерфейсу мережевого маршрутизатора. Що стосується порту SSH, то за замовчуванням він 22.

Після відкриття з'єднання SSH PuTTY відобразить консоль командного рядка. Оскільки система програмний засіб «Розумний будинок» захищена, консоль запитуватиме ім'я користувача та пароль. Після успішного входу в систему відобразиться екран привітання.

У меню інструментів MQTT можна ввімкнути на вкладці «Додаткові компоненти». MQTT використовується системою, побудованою в цій дипломній роботі, і його можна ввімкнути, вибравши в меню. Окрім MQTT, також було ввімкнено WIFI, і це було зроблено на вкладці «Налаштування системи».

4.6.2. Додавання пристроїв і служб

Кожен пристрій, підключений до програмного засобу «Розумний будинок», є різним і потребує базових компонентів, щоб представляти їх усіх. Основними базовими компонентами, відповідальними за додавання пристроїв і послуг,

керування ними, є додатки, речі та елементи. Кожен із цих базових компонентів виконує свою функцію.

Речі — це базові компоненти, що представляють усі сутності, як-от пристрої, служби тощо, якими керує система. Вони підключаються до системи через надбудови, які дозволяють системі керувати ними. Система може отримати доступ до функцій пристроїв або послуг через канали, які має кожна відповідна річ.

Елемент - базовий компонент, який може представляти всі властивості системи автоматизації. Це можуть бути рядки, числа, перемикачі, повзунки, палітри кольорів або інші типи елементів. Згадується, що предмет можна підключити до речі, щоб мати контроль над його каналом. Елементи можна використовувати як для автоматизації, так і для визначення інтерфейсів користувача, оскільки вони забезпечують взаємодію з речами за допомогою відповідних пристроїв і служб, до яких вони підключені.

По-перше, щоб додати пристрій до системи, потрібно встановити додатковий компонент для цього пристрою. Це можна зробити шляхом зміни addons.cfg файлу, додавши ім’я додаткового пакета до списку прив’язок, як зображено на рисунку 4.9.

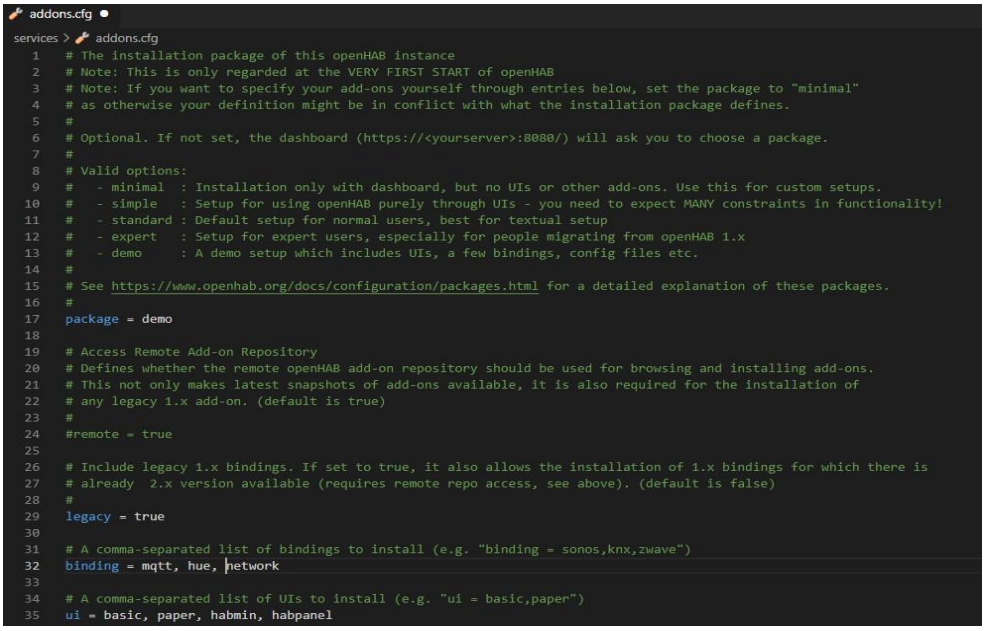


Рис. 4.9. addons.cfg файл з інструкціями та переліком встановлених доповнень.

У цій дипломній роботі, оскільки єдиними розумними пристроями, до яких ми мали доступ, були розумні світильники Philips Hue, доповнення називалося Philips Hue Binding із назвою пакета «hue». Для послуг ми використовували надбудову для MQTT з назвою пакета «mqtt». Його було додано до списку прив'язок yadd-ons.cfg.

Після ввімкнення підтримки пристроїв і служб для програмного засобу «Розумний будинок» шляхом встановлення відповідних доповнень наступним кроком є додавання пристроїв і служб. Це можна зробити, вказавши їх у файлі default.things,вручну.

У цій дипломній роботі пристрої були додані за допомогою використання Paper UI, оскільки цей процес здавався швидшим і простішим. Щоб додати пристрої в Paper UI, ми спочатку перейшли в меню «Вхідні» та натиснули кнопку «Сканувати», як показано на рисунку 4.10. Через кілька секунд почали з'являтися пристрої. Світильники Philips Hue, які ми мали, були підключені до Hue Bridge. І міст, і кожне світло було додано, натиснувши синє коло з білою галочкою, а потім кнопку «ДОДАТИ ЯК РІЧ», як показано на рисунку 4.11. та рисунку 4.12.

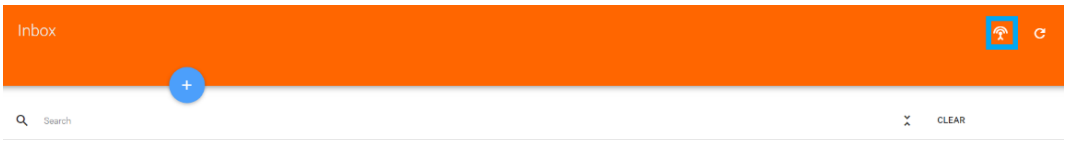


Рис. 4.10. Сканування пристроїв у мережі через меню «Вхідні».

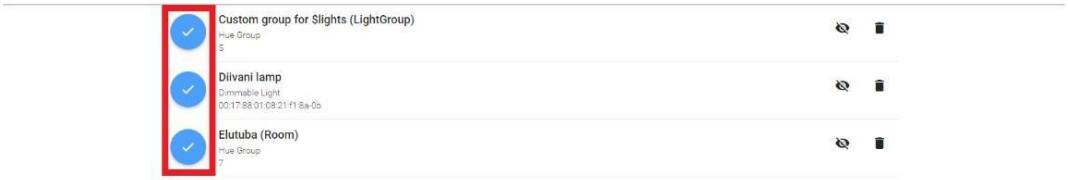


Рис. 4.11. Кнопки для додавання знайдених пристроїв у мережі.

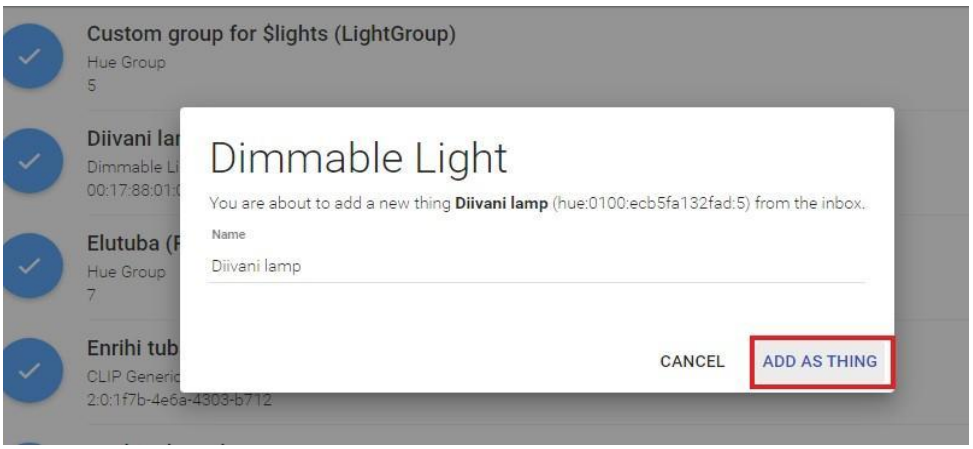


Рис. 4.12. Спливаюче вікно підтвердження додавання пристрою.

Сервіс MQTT було додано шляхом зміни файлу gsdefault.things і додавання додаткового файлу конфігурації mqtt.cfg. Цей файл конфігурації призначений для зберігання основних параметрів для Сервісу MQTT. Для сервісу MQTT визначено брокера та клієнта, де додано Канал для клієнта для зберігання даних із підписаної теми. На рисунку 4.13. і на рисунку 4.14. показано вміст описаних файлів.

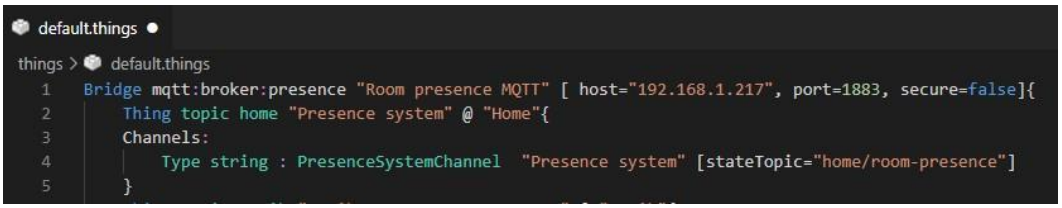


Рис. 4.13. Приклад конфігураційного файлу Mosquitto MQTT.

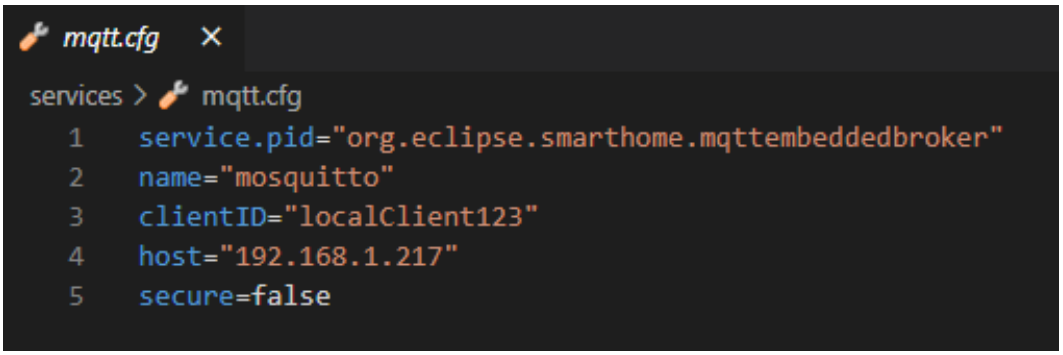


Рис. 4.14. MQTT, налаштований як indefault.things файл.

4.6.3. Автоматизація

Відповідно до документації програмний засіб «Розумний будинок», має базовий компонент, відомий як правила. Кожне правило працює через спрощений механізм правил і викликає сценарій під час запуску. Правила можна визначити за допомогою інтерфейсу користувача або шляхом запису файлів правил, де файл може містити кілька правил. Синтаксис правила базується на Xbase і відповідає структурі:

- Назва правила – унікальна назва для кожного правила.
- Умова тригера – подія, яка запускає виконання правила.
- Блок сценарію – контейнер для логіки, яка має бути виконаний на тригері. Для виконання правила існують різні категорії тригерів:
  - На основі подій елемента – тригери на основі елемента, реагують на оновлення елемента.
  - На основі подій - тригери на основі речей, реагують на зміни стану пристрою чи служби.
  - На основі групових подій – тригери на основі груп реагують на зміни стану елемента в певній групі.
  - Подія часу – тригери на основі часу, реагують у вказаний час.
  - На основі системних подій – системні тригери реагують на події запуску системи.

Автоматизація, реалізована в цій дипломній роботі, призначена для керування пристроями, які знаходяться в одній кімнаті з користувачем. По суті, оскільки єдиними розумними пристроями, які ми маємо для цієї дипломної роботи, є розумні світильники Philips Hue, тоді, якщо користувач перебуває поблизу кімнати, де є світло, тоді світло в цій кімнаті увімкнеться. Так само, якщо користувач залишає кімнату, світло в ній вимикається.

Ця автоматизація досягається за допомогою правила, яке запускається щоразу, коли клієнт MQTT, який підписаний на певну тему, отримує нові дані. Ці дані мають формат JSON і містять інформацію про:

- з якої кімнати надійшли ці дані;
- потужність сигналу Bluetooth між кожним пристроєм користувача та трекером;
- якщо в кімнаті є гості.

Це правило використовує дані, щоб визначити, чи хтось із користувачів переїхав до іншої кімнати, порівнюючи потужність сигналу між поточною та останньою кімнатами. Якщо поточна кімната відрізняється від попередньої кімнати та потужність сигналу краща, тоді користувача записують у нову кімнату. Це правило також виключає користувача з будь-якої кімнати, якщо його сигнал Bluetooth не може бути визначений.

Коли місцезнаходження всіх користувачів визначено, правило перевірить, чи кількість користувачів кімнати досягла нуля, або стала більшою за нуль. Якщо він досягає нуля, освітлення Philips Hue у цій кімнаті вимикається. Але якщо кількість користувачів змінюється від нуля до більшого значення, тоді в цій кімнаті вмикається світло.

Крім того, перемикач вимикає або вмикає це правило, яким користувачі можуть керувати вручну через інтерфейс користувача.

4.7. Конфігурація системи відстеження користувачів

У цьому розділі мова піде про систему відстеження користувачів. У наступних підрозділах буде описано огляд системи, процес налаштування та, нарешті, пояснено сценарій, який використовується для відстеження користувачів.

Система відстеження користувачів складатиметься з кількох окремих трекерів. Кожен трекер буде розміщено в різних кімнатах і незалежно відстежуватиме місцезнаходження користувача за допомогою сканування Bluetooth. Потім ця інформація надсилається на платформу домашньої автоматизації через MQTT. Щоб трекери могли виконувати такі дії, вони повинні мати підтримку WIFI і Bluetooth. Окрім цього, вони також повинні мати можливість запускати сценарій, який шукає

користувачів, компілює дані та надсилає їх через MQTT, як показано на рисунку 4.15.

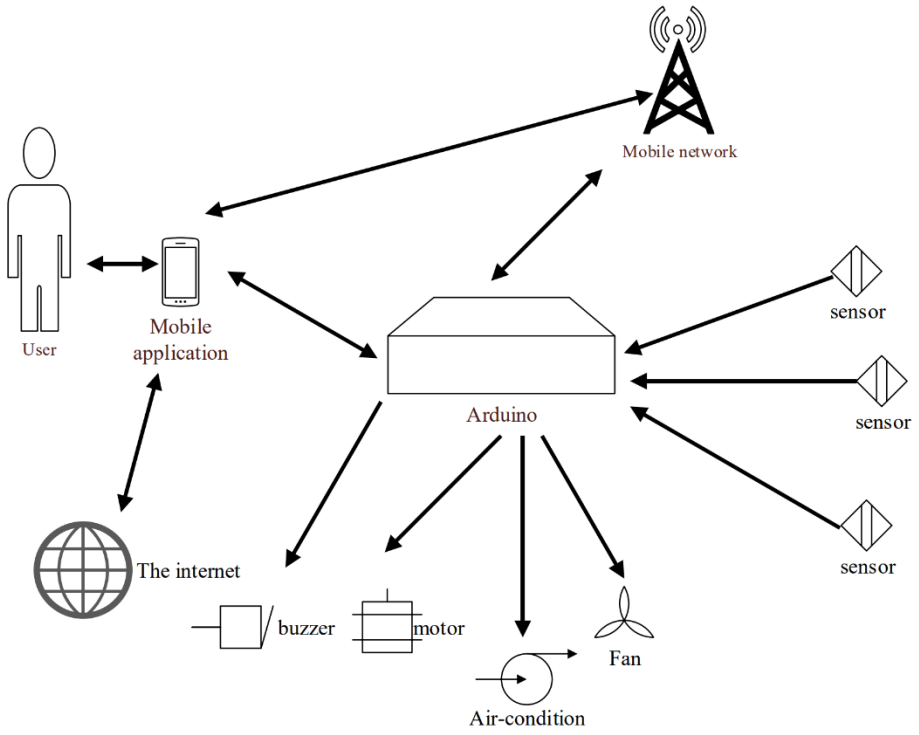


Рис. 4.15 Модель роботи системи відстеження користувачів

Як згадувалося в розділі «Довідкова інформація» цієї дипломної роботи, мікроконтролери Arduino та Raspberry Pi є одноплатними комп'ютерами, доступними для таких завдань. У цій дипломній роботі для кожного трекера буде використано Raspberry Pi Zero W. Основними причинами вибору Raspberry замість Arduino було те, що вони дозволяють краще експериментувати з різним програмним забезпеченням і мовами сценаріїв і є більш доступними.

Згідно з офіційною специфікацією продукту Raspberry, використовувані Raspberry Pi Zero W мають наступні характеристики:

- ЦП – Одноядерний, 1 ГГц
- Оперативна пам'ять – 4 ГБ LPDDR4-3200 SDRAM
- WIFI – бездротовий зв'язок 2,4 ГГц 802.11 b/g/n
- Bluetooth – Bluetooth 4.1 і BLE

Кожен Raspberry Pi Zero W працюватиме під управлінням Raspberry Pi OS Lite на базі Linux, яка є портом Debian. Ця операційна система не має графічного виводу.

Пропонована система має апаратну та програмну частини. Є онлайн-частина та GSM-частина системи. Таким чином, програмі потрібен Інтернет і мобільна мережа для виконання таких завдань, як відкривання/закривання дверей, вікон, освітлення, кондиціонування повітря та виконання частини захисту, наприклад відкриття/закриття водяного насоса, роботи сигналізації та вентилятора. Система підключається до Інтернету через кабель локальної мережі та мобільної мережі через SIM-карту

За допомогою обладнання та операційної системи, представлених у попередньому розділі, можна налаштувати кожен трекер.

Трекери, які були налаштовані в цій дипломній роботі, вимагали таких компонентів:

- Raspberry Pi Zero W
- Образ Raspberry Pi OS Lite
- Інструмент Flash
- Картка Micro-SD із пам’яттю 4 ГБ або більше
- Пристрій для читання карт Micro-SD
- WIFI доступ

Щоб запустити Raspberry Pi OS Lite на Raspberry Pi Zero W, спочатку потрібен файл зображення. Для цього такий інструмент, як Raspberry Pi Imager можна використовувати для отримання файлу зображення та перезавантаження зображення на картку Micro-SD. Як і у випадку з налаштуванням платформи домашньої автоматизації, щоб записати зображення на картку Micro-SD, потрібен пристрій для читання карт Micro-SD.

Після того, як зображення отримано та завантажено на картку Micro-SD, зображення має бути налаштовано для підключення до WIFI після завантаження та дозволу підключення SSH. Для цього потрібно отримати доступ до кореневого каталогу зображення. Для ввімкнення SSH порожній файл з назвою ssh має бути створено. Що стосується ввімкнення доступу до WIFI, файл під назвою

wpa_supplicant.conf, має бути створено з наступним вмістом, який можна знайти на рисунку 4.16.

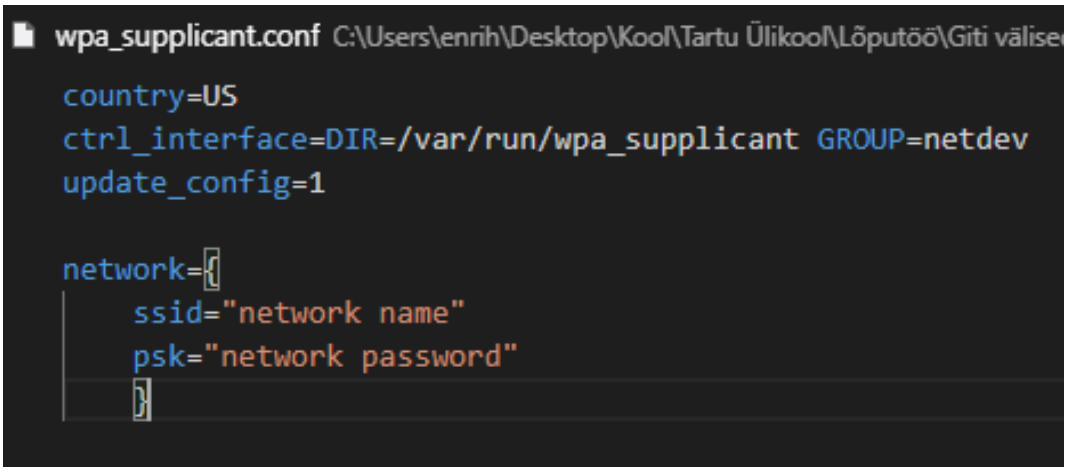


Рис. 4.16. Приклад wpa_supplicant.conf файлу.

Після завершення налаштування для SSH і WIFI карту Micro-SD можна встановити в Raspberry Pi Zero W і завантажити. Для повного завантаження трекерів може знадобитися кілька хвилин, але коли вони з’являться у списку пристроїв мережевих маршрутизаторів, можна створити сеанс SSH для підключення до трекерів.

Коли сеанс SSH буде створено, консоль запропонує користувачеві увійти. Ім’я користувача за замовчуванням — «pi», а пароль — «raspberry». Після цього отримується доступ до системи Raspberry Pi OS. Оскільки Raspberry Pi OS є системою на базі Linux, консоль командного рядка може працювати за допомогою команд Linux.

Щоб скрипт відстеження користувачів працював, необхідно встановити додаткові пакети. Після цього необхідно встановити модуль із інструментами Bluetooth за допомогою такої команди: sudo apt-get install bluetooth bluez libbluetooth-dev.

Тепер, коли встановлені модулі для передачі даних через MQTT і роботи з Bluetooth, потрібні інструменти, необхідні для запуску скрипта Python. Цей сценарій Python використовуватиме paho-mqtt модуль, що дозволяє використовувати ресурси

MQTT ipybluez модуль для увімкнення використання ресурсів Bluetooth у сценарії. Python і згадані модулі Python можна встановити за допомогою таких команд:

- sudo apt-get встановити python-pip
- sudo python -m pip встановити paho-mqtt
- sudo python -m pip встановити pybluez

У сценарії відстеження користувачів визначено кілька завдань. Ці завдання пов’язані з обробкою служби MQTT, скануванням пристроїв Bluetooth, вимірюванням потужності їхнього сигналу від трекера, а також збиранням і надсиланням отриманих даних через MQTT.

MQTT має завдання для обробки підключення та відключення. У разі відключення від функції брокера, обробник сигналу буде запущено, що закриє клієнт MQTT і завершить роботу сценарію. Що стосується роботи з підключенням, функції on_connect спрацьовує під час спроби підключення до брокера. Після активації функція роздрукує код результату підключення. Ця функція може бути використана в майбутньому для запуску певних дій, але наразі не має жодних функцій.

Існують функції для пошуку пристроїв Bluetooth і вимірювання потужності їх сигналу. Перша функція намагатиметься виявити нові пристрої, і якщо знайдено пристрій із достатньо сильним сигналом, функція повертає значення true, інакше — false. Друга функція шукатиме попередньо визначений пристрій і вимірюватиме рівень його сигналу, який буде повернено в кінці.

Основна функція працює, спочатку налаштувавши MQTT і підключившись до брокера на платформі домашньої автоматизації. Після цього система шукатиме гостей протягом певного часу, а потім отримуватиме потужність сигналу від кожного попередньо визначеного пристрою користувача. Коли це буде зроблено, дані будуть упаковані у формат JSON і надіслані дією публікації MQTT. Після цього цикл повторює процес.

Робочий процес системи «розумний дім» від визначення місцезнаходження користувача до керування інтелектуальними пристроями вдома за допомогою

автоматизації пояснюється на рисунку 4.17. Робочий процес складається з чотирьох основних груп, кожна з яких містить один або кілька другорядних компонентів.

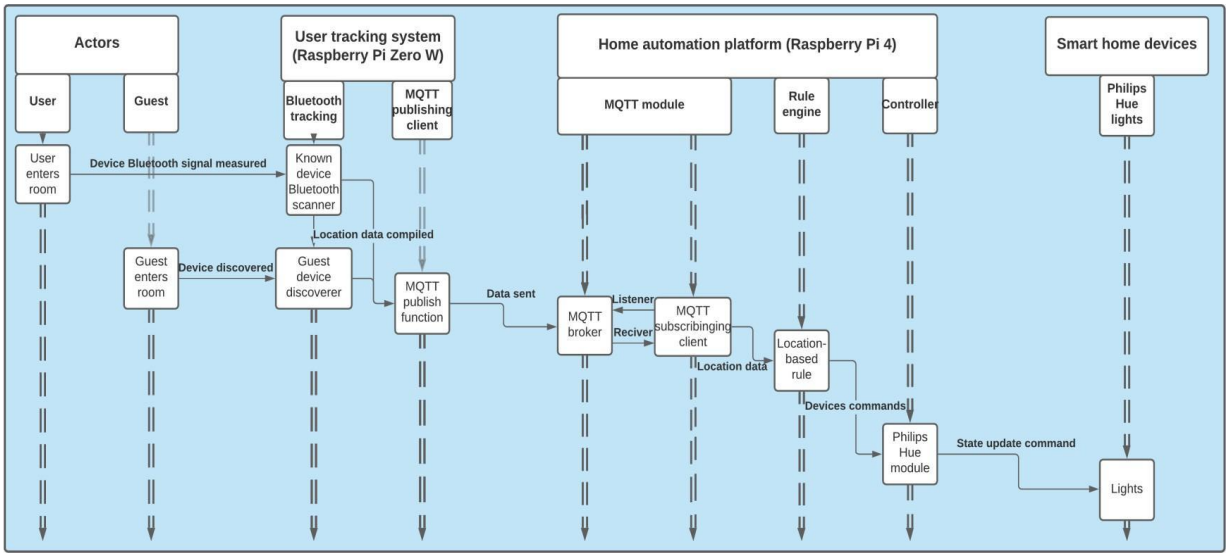


Рис. 4.17. Діаграма послідовності робочого процесу системи розумного дому.

Перша група – Актори. Ця група містить різних акторів, які взаємодіють із системою. Вони є користувачами та гостями, можуть взаємодіяти з системою, переміщаючись по дому, з однієї кімнати в іншу.

Друга група – система стеження за користувачами. Ця діаграма представляє окремі трекери в різних кімнатах. Ця група складається з двох компонентів: модуля

відстеження Bluetooth і клієнта публікації MQTT. Модуль відстеження Bluetooth сканує пристрої поблизу та виявляє відомі та невідомі пристрої поблизу від користувачів і гостей. Потужність сигналу кожного виявленого пристрою вимірюється та збирається в дані про місцезнаходження. Ці дані також містять інформацію про гостей і кімнату, в якій знаходиться трекер. Після того, як дані скомпільовані, клієнт публікації MQTT надсилає дані брокеру систем домашньої автоматизації MQTT.

Третя група — платформа домашньої автоматизації. Ця група містить три компоненти: модуль MQTT, механізм правил і контролер. Модуль MQTT відповідає за роботу з брокером MQTT і підписаним клієнтом. Клієнт MQTT, який підписався, слухає брокера MQTT і отримує нові дані. Звідти дані пересилаються до механізму правил.

Механізм правил відповідає за обробку автоматизації та може містити кілька правил. Правило автоматизації на основі розташування, реалізоване в цій дипломній роботі, запускається щоразу, коли клієнт MQTT, який підписався, отримує нові дані. Правило приймає отримані дані та обробляє їх. Після цього він використовуватиме ці дані, щоб визначити, які пристрої слід вимкнути або ввімкнути, і надішле цю інформацію на контролер. Контролер використовує ці дані для надсилання команд кожному відповідному пристрою.

Висновок

У даному розділі описано реалізацію програмного прототипу, який демонструє запропонований підхід до автоматизованого збору, обробки та аналізу даних для оцінки енергоефективності системи «Розумний будинок».

Прототип реалізовано у вигляді клієнт-серверного додатку з веб-інтерфейсом. Серверна частина виконує функції збору даних з датчиків, розрахунку цільових показників ефективності, формування звітності. Клієнтська частина надає користувачу інструменти візуалізації та аналізу даних.

Розроблений прототип дозволяє продемонструвати життєздатність та практичне застосування запропонованих теоретичних та методологічних рішень в сфері оцінки енергоефективності систем класу "Розумний будинок"

ВИСНОВОК

У результаті дипломного проектування розроблено комплексну методику оцінки та контролю енергоефективності для інтелектуальних систем управління об'єктами нерухомості класу «Розумний будинок». Методика базується на вимірюванні профільних показників енергоспоживання та їх аналізі з використанням сучасного математичного апарату.

Реалізовано програмний засіб для автоматизації запропонованої методики. Програмне рішення дозволяє автоматизувати процеси збору, обробки, аналізу даних, генерування звітності та надання рекомендацій для прийняття рішень щодо підвищення рівня енергоефективності системи.

Впровадження результатів дипломного проектування дозволить підвищити енергоефективність та оптимізувати енергоспоживання в системах «Розумний будинок»

Ця дипломна робота була спрямована на створення системи розумного будинку, яка могла б контролювати розумну домашню техніку. У результаті цієї дипломної роботи була створена програмний засіб системи «Розумний будинок» для керування цими пристроями та досягнення автоматизації за допомогою даних, які надсилаються через протокол зв’язку. Ця система складалася з двох підсистем: платформи домашньої автоматизації та системи відстеження користувачів.

Платформа домашньої автоматизації була підсистемою, відповідальною за управління будинком шляхом моніторингу, контролю та автоматизації пристроїв і послуг розумного будинку. Цю платформу було реалізовано за допомогою системи розумного будинку. Він дозволив підключати та керувати кількома різними пристроями та службами з різних екосистем і працює на одноплатних комп’ютерах, таких як Raspberry Pi. Таким чином, Raspberry Pi 4 використовувався для запуску платформи домашньої автоматизації.

Для демонстрації можливостей програмного засобу у середовищі розумного дому було використано дружній до Інтернету речей протокол обміну повідомленнями MQTT, щоб забезпечити взаємодію двох підсистем. Платформа

домашньої автоматизації містила брокера Mosquitto MQTT і абонентського клієнта, де абонентський клієнт реєстрував дані про місцезнаходження користувача, які надсилалися через мережу IoT. Ці дані використовувалися правилом автоматизації для вмикання та вимикання світильників Philips Hue залежно від близькості користувачів до них.

Щоб визначити місцезнаходження користувачів у будинку, система відстеження користувачів, яка була розроблена та створена в цій дипломної роботі, складалася з окремих пристроїв відстеження, розміщених у різних кімнатах навколо будинку. Ці пристрої відстеження були створені за допомогою Raspberry Pi Zero W зі сценарієм відстеження користувачів. Ця модель одноплатних комп'ютерів Raspberry Pi може використовувати WIFI і Bluetooth і запускати легку операційну систему. Ця операційна система розміщувала службу Mosquitto MQTT і запускала сценарій Python, відповідальний за збір даних Bluetooth і публікацію їх через клієнт публікації MQTT.

Обидві підсистеми створюють цілісну систему розумного дому IoT, яка відстежує та контролює розумну домашню техніку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. ГОСТ 19.003-80 ЄСПД. Схеми алгоритмів і програм. Позначення умовні графічні.
2. ГОСТ 19.105-78 ЄСПД. Загальні вимоги до програмних документів.
3. ГОСТ 19.401-78 ЄСПД. Текст програми. Вимоги до змісту і оформлення.
4. Хахаев И.А. Практикум по алгоритмизации и программированию на *PHP*. – М.: Альт Линукс, 2010. — 126 с.
5. ДСТУ 3008-95 Документація. Звіти у сфері науки і техніки. Структура і правила оформлення.
6. НД ТЗІ 1.1-003-99. Термінологія у області захисту інформації в комп'ютерних системах від несанкціонованого доступу. // Департамент спеціальних телекомунікаційних систем і захисту інформації Служби безпеки України. – Київ, 1999.
7. Бойченко С.В., Іванченко О.В. Положення про дипломні роботи (проекти) випускників Національного авіаційного університету. – К.: НАУ, 2017. – 63 с.
8. Лутц М. Изучаем Python, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 1280 с.
9. Златопольский Д.М. Основы программирования на языке Python. – М.: ДМК Пресс, 2017. – 284 с.
10. Лутц М. Программирование на Python, том I, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 992 с.
11. Любанович Билл Простой Python. Современный стиль программирования. – СПб.: Питер, 2016. – 480 с.
12. Доусон М. Программируем на Python. – СПб.: Питер, 2014. – 416 с.
13. Leitão, J.; Gil, P.; Ribeiro, B.; Cardoso, A. A survey on home energy management. IEEE Access 2020, 5722 p.
14. Yahia, Z.; Pradhan, A. Optimal load scheduling of household appliances considering consumer preferences: An experimental analysis. Energy 2018, 163 p.

15. Singh, S.; Roy, A.; Selvan, M. Smart load node for nonsmart load under smart grid paradigm: A new home energy management system. IEEE Consum. Electron. Mag. 2019, 27 p.

16. Zafari, F.; Papapanagiotou, I.; Christidis, K. Microlocation for Internet-of-Things-equipped smart buildings. IEEE Internet Things J. 2016, 112 p.

17. Alam, M.R.; Reaz, M.B.I.; Ali, M.A.M. A review of smart homes—Past, present, and future. IEEE Trans. Syst. Man Cybern. C Appl. Rev. 2012, 1203 p.

18. Lin, Y.H. Trainingless multi-objective evolutionary computing-based nonintrusive load monitoring: Part of smart-home energy management for demand-side management. J. Build. Eng. 2021, 601 p.

19. Wang, X.; Mao, X.; Khodaei, H. A multi-objective home energy management system based on internet of things and optimization algorithms. J. Build. Eng. 2021,1603 p.

20. Gholinejad, H.R.; Loni, A.; Adabi, J.; Marzband, M. A hierarchical energy management system for multiple home energy hubs in neighborhood grids. J. Build. Eng. 2020, 1028 p.

21. Khemakhem, S.; Rekik, M.; Krichen, L. A collaborative energy management among plug-in electric vehicle, smart homes and neighbors' interaction for residential power load profile smoothing. J. Build. Eng. 2020, 976 p.

22. Li, W.J.; Tan, X.; Tsang, D.H.K. Smart home energy management systems based on non-intrusive load monitoring. In Data Management, Grid Analytics, and Dynamic Pricing, Proceedings of the 2015 IEEE International Conference on Smart Grid Communications (SmartGridComm), Miami, FL, USA, 2–5 November 2015; IEEE: New York, NY, USA, 2015 p.

23. Lee, S.; Choi, D.H. Reinforcement learning-based energy management of smart home with rooftop solar photovoltaic system, energy storage system, and home appliances. Sensors 2019, 3937 p.

24. Ran, X.; Leng, S. Enhanced robust index model for load scheduling of a home energy local network with a load shifting strategy. IEEE Access 2019, 19953 p.

25. Lin, Y.H.; Hu, Y.C. Residential consumer-centric demand-side management based on energy disaggregation-piloting constrained swarm intelligence: Towards edge computing. Sensors 2018, 1365 p.

26. Veras, J.M.; Silva, I.R.S.; Pinheiro, P.R.; Rabêlo, R.A.L.; Veloso, A.F.S.; Borges, F.A.S.; Rodrigues, J.J.P.C. A multi-objective demand response optimization model for scheduling loads in a home energy management system. Sensors 2018, 3207 p.

27. Soetedjo, A.; Nakhoda, Y.I.; Saleh, C. Embedded fuzzy logic controller and wireless communication for home energy management systems. Electronics 2018, 189 p.

28. Aghajani, G.; Ghadimi, N. Multi-objective energy management in a micro-grid. Energy Rep. 2018, 225 p.

29. Rasheed, M.B.; Javaid, N.; Ahmad, A.; Jamil, M.; Khan, Z.A.; Qasim, U.; Alrajeh, N. Energy optimization in smart homes using customer preference and dynamic pricing. Energies 2016, 593 p.

30. Lin, Y.H.; Tsai, M.S. An advanced home energy management system facilitated by nonintrusive load monitoring with automated multiobjective power scheduling. IEEE Trans. Smart Grid 2015, 1851 p.

31. Palensky, P.; Dietrich, D. Demand side management: Demand response, intelligent energy systems, and smart loads. IEEE Trans. Ind. Informat. 2011, 388 p.

32. Hosseinnezhad, V.; Shafie-Khah, M.; Siano, P.; Catalão, J.P.S. An optimal home energy management paradigm with an adaptive neuro-fuzzy regulation. IEEE Access 2020, 19628 p.

33. Kong, W.; Dong, Z.Y.; Ma, J.; Hill, D.J.; Zhao, J.; Luo, F. An extensible approach for non-intrusive load disaggregation with smart meter data. IEEE Trans. Smart Grid 2018, 3372 p.

34. He, J.; Zhang, Z.; Zhu, L.; Zhu, Z.; Liu, J.; Gai, K. An efficient and accurate nonintrusive load monitoring scheme for power consumption. IEEE Internet Things J. 2019, 9063 p.

35. Xiao, Y.; Hu, Y.; He, H.J.; Zhou, D.G.; Zhao, Y.; Hu, W.S. Non-intrusive load identification method based on improved KM algorithm. IEEE Access 2019, 151377 p.

36. Ruano, A.; Hernandez, A.; Ureæa, J.; Ruano, M.; Garcia, J. NILM techniques for intelligent home energy management and ambient assisted living: A review. Energies 2019, 2203 p.

37. Wu, X.; Jiao, D.; Du, Y. Automatic implementation of a self-adaption non-intrusive load monitoring method based on the convolutional neural network. Processes 2020, 704 p.

38. Shi, W.; Cao, J.; Zhang, Q.; Li, Y.; Xu, L. Edge computing: Vision and challenges. IEEE Internet Things J. 2016, 646 p.

39. Marah, B.D.; Jing, Z.; Ma, T.; Alsabri, R.; Anaadumba, R.; Al-Dhelaan, A.; Al-Dhelaan, M. Smartphone architecture for edge-centric IoT analytics. Sensors 2020, 892 p.

40. Le, T.-T.-H.; Kang, H.; Kim, H. Household appliance classification using lower odd-numbered harmonics and the bagging decision tree. IEEE Access 2020, 55952 p.

41. Schirmer, P.A.; Mporas, I.; Sheikh-Akbari, A. Energy disaggregation using two-stage fusion of binary device detectors. Energies 2020, 2148 p.

42. Lu, M.; Li, Z. A hybrid event detection approach for non-intrusive load monitoring. IEEE Trans. Smart Grid 2020, 540 p.

43. Le, T.-T.-H.; Kim, H. Non-intrusive load monitoring based on novel transient signal in household appliances with low sampling rate. Energies 2018, 3409 p.

44. Puente, C.; Palacios, R.; González-Arechavala, Y.; Sánchez-Úbeda, E.F. Non-intrusive load monitoring (NILM) for energy disaggregation using soft computing techniques. Energies 2020, 3117p.

45. Mengistu, M.A.; Girmay, A.A.; Camarda, C.; Acquaviva, A.; Patti, E. A cloud-based on-line disaggregation algorithm for home appliance loads. IEEE Trans. Smart Grid 2018, 3439 p.

46. Wójcik, A.; Łukaszewski, R.; Kowalik, R.; Winiecki, W. Nonintrusive appliance load monitoring: An overview, laboratory test results and research directions. Sensors 2019, 3621 p.

47. Atlam, H.F.; Walters, R.J.; Wills, G.B. Fog computing and the Internet of Things: A review. Big Data Cogn. Comput. 2018, 10 p.

48. Sudqi Khater, B.; Abdul Wahab, A.W.B.; Idris, M.Y.I.B.; Abdulla Hussain, M.; Ahmed Ibrahim, A. A lightweight perceptron-based intrusion detection system for fog computing. Appl. Sci. 2019, 178p.

49. Wang, T.; Ke, H.; Zheng, X.; Wang, K.; Sangaiah, A.K.; Liu, A. Big data cleaning based on mobile edge computing in industrial sensor-cloud. IEEE Trans. Ind. Inform. 2020, 1329 p.

50. Yigit, M.; Gungor, V.C.; Baktir, S. Cloud computing for smart grid applications. Comput. Netw. 2014, 329 p.

51. Abdullah, M.; Faruque, A.; Vatanparvar, K. Energy management-as-a-service over fog computing platform. IEEE Internet Things J. 2015, 169 p.

52. Chiang, M.; Zhang, T. Fog and IoT: An overview of research opportunities. IEEE Internet Things J. 2016, 864 p.

53. Hussain, M.M.; Beg, M.S. Fog computing for Internet of Things (IoT)-aided smart grid architectures. Big Data Cogn. Comput. 2019, 8 p.

54. Niagara Tridium (Tridium, Inc.). Available online: https://www.tridium.com/.

55. First International Computer, Inc. (FIC). Available online: https://www.fic.com.tw/.

56. ICP DAS CO., LTD. Available online: http://oldweb.icpdas.com/index.php.

57. Chen, Y.Y.; Lin, Y.H. A smart autonomous time- and frequency-domain analysis current sensor-based power meter prototype developed over fog-cloud analytics for demand-side management. Sensors 2019, 4443 p.

ДОДАТОК А. Текст програми

```
import asyncio

from OpenHub.homekit_accessories.homkit_sensor_interface
import HomeKitSensorInterface
import logging

from pyhap.const import CATEGORY_SWITCH

logger = logging.getLogger(__name__)


class Relay(HomeKitSensorInterface):
    run_debug_message = "Relay State: "

    def __init__(self, serial_no=None, display_name=None,
channel_interface_serial_no=None, *args, **kwargs):
        self.category = CATEGORY_SWITCH

        super().__init__(serial_no=serial_no,
display_name=display_name,

channel_interface_serial_no=channel_interface_serial_no,
*args, **kwargs)

    def set_display_name(self, display_name):
        if display_name is None:
            return "Relay"
        else:
            return display_name

    def add_functional_service(self):
        return self.add_preload_service('Relay')

    def add_functional_service_characteristic(self):
        return self.service.configure_char(
            'On', setter_callback=self.set_relay)

    def __setstate__(self, state):
        self.__dict__.update(state)
        # self._gpio_setup(self.pin)

    def set_relay(self, value):
```

```
        if value:
            asyncio.create_task(self.channel.turn_on())
        else:
            asyncio.create_task(self.channel.turn_off())

    async def stop(self):
        await self.channel.turn_off()
        await super().stop()

    async def run(self):
        pass
from abc import ABC, abstractmethod
from pyhap.accessory import Accessory
from pyhap.accessory import CATEGORY_OTHER
import logging
from OpenHub.globals import id_channels_map \
    , driver, accessory_id_data_transformer_map
import uuid


class HomeKitSensorInterface(ABC, Accessory):
    logger = logging.getLogger(__name__)
    category = CATEGORY_OTHER
    scale = None
    index = None
    service = None
    char = None

    serial_no = None
    display_name = None

    channel = None

    raw_value_converter = None

    run_debug_message = "Run Debug Message Not Implemented"

    calibrator = None
    aid = None

    data_transformer = None

    def __init__(self, serial_no=uuid.uuid4(),
display_name=None, channel_interface_serial_no=None,
```

```
                data_transformer=None,config=None, *args,
**kwargs):
        ABC.__init__(self)
        self.display_name =
self.set_display_name(display_name)
        self.serial_no = serial_no
        self.channel_serial_no = channel_interface_serial_no
        self.channel =
id_channels_map[str(self.channel_serial_no)]
        Accessory.__init__(self, driver=driver,
display_name=self.display_name,
                           *args, **kwargs)

        self.service = self.add_functional_service()
        self.char =
self.add_functional_service_characteristic()
        if config is not None and 'datatransformer' in
config.keys():
            self.data_transformer =
config['datatransformer']
        if config is not None and 'data_transformer' in
config.keys():
            self.data_transformer =
config['data_transformer']


    def add_info_service(self):
        serv_info =
self.driver.loader.get_service("AccessoryInformation")
        serv_info.configure_char("Name",
value=self.display_name)
        serv_info.configure_char("SerialNumber",
value=self.serial_no)
        serv_info.configure_char("Manufacturer",
value="BellyFrito")
        serv_info.configure_char("Model", value="DEFAULT")
        self.add_service(serv_info)

    @abstractmethod
    def set_display_name(self, display_name):
        pass

    @abstractmethod
    def add_functional_service_characteristic(self):
```

```
        pass

    @abstractmethod
    def add_functional_service(self):
        pass

    async def run(self):
        if self.data_transformer is None:
            data = await self.channel.run()
            self.logger.info(self.display_name + " Output: "
+ str(data))
            if 'averaged' in data.keys():
                if self.scale is None:

self.char.set_value(float(data['averaged']))
                else:

self.char.set_value(self.scale*float(data['averaged']))
            elif 'value' in data.keys():
                if self.scale is None:

self.char.set_value(float(data['value']))
                else:
                    self.char.set_value(self.scale *
float(data['value']))
        else:
            val = await self.data_transformer.run()
            self.logger.info(self.display_name + " Output: "
+ str(val))

            self.char.set_value(float(val))

import json
import logging
import RPi.GPIO as GPIO
from serial import Serial
from .channels.pi_pico_pump import PiPicoPump
from .channels.pi_pico_analog import PiPicoAnalog
from .channels.pi_pico_relay import PiPicoRelay
from .hardware_interface import HardwareInterface

logger = logging.getLogger(__name__)
from asyncio import Lock
```

```
class PiPico(HardwareInterface):
    lock = Lock()

    serial_no = None
    name = None

    serial_com = None
    serial = None
    interrupt = None

    channels = []

    def __init__(self, pico_config, channels=None, *args,
**kwargs):
        self.config = pico_config

        # self.channels_config = pico_config['channels']
        self.serial_no = pico_config['id']
        self.interrupt = pico_config['pi_gpio_interrupt']
        # if 'serial_com' in pico_config.keys():
        #     self.serial_com = pico_config['serial_com']
        # if self.serial_com is not None:
        #     self.serial = Serial(self.serial_com, 9600,
timeout=1)
        super(PiPico, self).__init__(self.serial_no,
channels, *args, **kwargs)

    def set_serial_com(self, serial_com):
        self.serial = serial_com

    def create_channel(self):
        for channel_config in self.channels_config.values():
            if 'type' == 'pump':
                self.channels.append(PiPicoPump(self, ))
            elif 'type' == 'relay':
                self.channels.append(PiPicoRelay(self, ))

    async def send_command(self, command, channel=None,
state=None):
        response = await self.send_command_lock(command,
channel, state, self.lock)
        return response
```

```
    async def send_command_lock(self, command, channel=None,
state = None, lock=None):
        command_json =
self.build_command_json(command,channel,state)
        sensor_response = b'{}'
        await lock.acquire()
        try:
            self.serial.flush()
            self.serial.write((json.dumps(command_json) +
'\n').encode('utf-8'))
            pico_data = self.serial.readline()
            sensor_response = pico_data[:-2]
            logger.info(sensor_response)
            self.serial.flush()
        except Exception as error:
            logger.error(str(error.with_traceback()))
        finally:
            lock.release()
            return sensor_response

    def build_command_json(self, command, channel,state):
        command_json = {'command': command}
        if channel is not None:
            command_json['channel'] = channel
        if state is not None:
            command_json['state'] = state
        return command_json

    async def initialize(self):
        command = 'init'
        return await self.send_command(command)
import logging
from abc import ABC, abstractmethod
import json
from OpenHub.globals import id_channels_map,id_stats_map

class DataTransformer(ABC):
    logger = logging.getLogger(__name__)

    homekit_accessory_serial_no = None

    def __init__(self, dct, data_transformers=None):
        self.logger.info(str(dct))
        self.channels = []
```

```
        self.stats=[]
        self.constants = dct['data_transformer_constants']

        for channel in dct['channels']:
            self.channels.append(id_channels_map[channel])
        for stat in dct['channel_stats']:
            self.logger.info('stats map')
            self.logger.info(str(id_stats_map))
            if stat in id_stats_map:
                self.logger.info(id_stats_map[str(stat)])
                self.stats.append(id_stats_map[str(stat)])
        self.data_transformers = dct['children']

        super().__init__()

    async def run(self):
        self.logger.info('datatransformer running')
        outputs = []
        if self.data_transformers is not None:
            for transformer in self.data_transformers:
                outputs.append(await transformer.run())
        if self.channels is not None:
            for channel in self.channels:
                channel_out = await channel.run()
                if 'averaged' in channel_out.keys():
                    channel_out =
float(channel_out['averaged'])
                elif 'value' in channel_out.keys():
                    channel_out =
float(channel_out['value'])
                outputs.append(channel_out)
        if self.stats is not None:
            for stat in self.stats:
                outputs.append(stat.value)
        if self.constants is not None:
            for constant in self.constants:
                outputs.append(constant.value)
        self.logger.info('datatransformer map ' +
str(outputs))

        return self.perform_op(outputs)

    @abstractmethod
    def perform_op(self, inputs):
```

```
        pass
import logging
from .data_transformer import DataTransformer


class Inverse(DataTransformer):

    def perform_op(self, inputs):
        product = 1
        for input in inputs:
            if input == 0:
                self.logger.info('input is 0 returning -1 ')

                return -1
            product = float(1/float(input))
        self.logger.info('inverted ' + str(product))
        return product
import logging
from .data_transformer import DataTransformer


class Sum(DataTransformer):

    def perform_op(self, inputs):
        out = sum(inputs)
        self.logger.info('summed ' + str(out))

        return out

import logging
from .data_transformer import DataTransformer


class Product(DataTransformer):

    def perform_op(self, inputs):
        product = 1
        for input in inputs:
            product = product * input
        self.logger.info('product ' + str(product))

        return product
```