МАГІСТЕРСЬКА РОБОТА

МР. ІПм - 19.00.00.000 ПЗ

Група ІПм-23-3

Перегінець Максим

2024

Івано-Франківський національний технічний університет нафти і газу

Інститут інформаційних технологій

Кафедра інженерії програмного забезпечення

Перегінець Максим Володимирович

(прізвище, ім'я, по батькові)

УДК 004.942

(індекс)

МАГІСТЕРСЬКА РОБОТА

Імплементація концептуального фреймворку Інтернету речей для обміну даними в інтелектуальних мережах

(назва роботи)

Інженерія програмного забезпечення

(назва освітньої програми)

121 - Інженерія програмного забезпечення

(шифр і назва спеціальності)

Перегінець М.В.

(підпис, ініціали та прізвище здобувача освітнього ступеня)

Науковий керівник Вовк Роман Богданович, к.т.н., доцент

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

Допущено до захисту

Завідувач кафедри

доц. Бандура В.В.

(посада) (підпис) (дата) (ініціали та прізвище)

Нормоконтроль

доц. Вовк Р.Б.

(посада) (підпис) (дата) (ініціали та прізвище)

Робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Івано-Франківськ – 2024

Івано-Франківський національний технічний університет нафти і газу

Інститут інформаційних технологій

Кафедра інженерії програмного забезпечення

Освітній рівень магістр

Спеціальність 121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:
Зав. кафедрою **ІПЗ**
доц. В.В. Бандура
“ 04 ” вересня 2024 р.

ЗАВДАННЯ

НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Перегінцю Максиму Володимировичу

(прізвище, ім’я, по-батькові)

1. Тема магістерської роботи “ Імплементація концептуального фреймворку Інтернету речей для обміну даними в інтелектуальних мережах”

керівник проекту (роботи) Вовк Роман Богданович, к.т.н., доцент

затверджені наказом закладу вищої освіти від “ 22 ” листопада 2024 р. № 781/7

2. Строк подання студентом проекту (роботи) 15 грудня 2024 р.

3. Вихідні дані до проекту (роботи) Теоретичні концепції та формальні моделі побудови та функціонування інформаційних та програмних технологій Інтернету речей

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

1. Дослідження предметної області функціонування інтелектуальних мереж та Інтернету речей
2. Методологічні основи застосування Інтернету речей в контексті інтелектуальних мереж
3. Дослідження концепцій розумного будинку, розумної будівлі та розумної мережі
4. Імплементація фреймворку Інтернету речей для обміну даними в інтелектуальних мережах

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

1. Загальна архітектура IoT (рис. 1.1)
2 Концептуальна модель розумної мережі та її відповідні стандарти (рис. 1.2)
3. Рівень зв'язку в розумній мережі (рис. 1.3)
4. Архітектура черги повідомлень Advanced Message Queue Telemetry Transport (рис. 1.4)
5. Загальна архітектура посередника запитів об’єктів (рис. 1.5)

6. Консультанти розділів проекту (роботи)

Розділ	Консультант	Підпис, дата
Перевірка на плагіат	доц., к.т.н. Вовк Р.Б.	

7. Дата видачі завдання 04 вересня 2024 р.

Керівник _____
(підпис)

Завдання прийняв до виконання_____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назви етапів магістерської роботи	Строк виконання етапів роботи	Примітка
1	Підбір і вивчення літератури по темі магістерської роботи	15.09.2024	виконано
2	Аналіз концепцій та алгоритмів предметної області	29.09.2024	виконано
3	Дослідження предметної області функціонування інтелектуальних мереж та Інтернету речей	15.10.2024	виконано
4	Методологічні основи застосування Інтернету речей в контексті інтелектуальних мереж	08.11.2024	виконано
5	Дослідження концепцій розумного будинку, розумної будівлі та розумної мережі	20.11.2024	виконано
6	Імплементація фреймворку Інтернету речей для обміну даними в інтелектуальних мережах	01.12.2024	виконано
7	Затвердження пояснювальної записки роботи завідувачем кафедри	15.12.2024	виконано

Студент – магістр _____
(підпис)

Керівник роботи _____
(підпис)

АНОТАЦІЯ

Магістерська робота: 83 с., 27 рис., 54 джерела.

Тема: Імплементація концептуального фреймворку Інтернету речей для обміну даними в інтелектуальних мережах

Об'єкт дослідження: процеси обміну даними в інтелектуальних мережах, інтегрованих із системами Інтернету речей.

Мета роботи: розробка та імплементація фреймворку Інтернету речей для обміну даними в інтелектуальних мережах, що забезпечує підвищення ефективності обміну інформацією між IoT-пристроями та системами управління, а також оптимізацію використання системних ресурсів.

Предмет дослідження: методи та технології розробки фреймворку IoT для забезпечення ефективного обміну даними в інтелектуальних мережах, включаючи підходи до проектування системи, на основі мікросервісів.

Результати дослідження

У ході дослідження представлено фреймворк для інтеграції IoT із інтелектуальними мережами, що відрізняється підвищеною ефективністю обміну даними та можливістю масштабування завдяки використанню мікросервісної архітектури.

Висновок

Результати дослідження можуть бути використані для впровадження IoT-фреймворків у розумних мережах різного призначення, зокрема у сферах енергетики, розумних міст, автоматизації виробничих процесів та управління інфраструктурою.

ІНТЕРНЕТ РЕЧЕЙ (IOT), ІНТЕЛЕКТУАЛЬНІ МЕРЕЖІ, ОБМІН ДАНИМИ, МІКРОСЕРВІСНА АРХІТЕКТУРА, ОПТИМІЗАЦІЯ, ПРОТОКОЛИ IOT, ЕФЕКТИВНІСТЬ СИСТЕМ, АВТОМАТИЗАЦІЯ, РОЗУМНІ СЕРЕДОВИЩА.

ABSTRACT

Master Thesis: 83 pp., 27 fig., 54 sources.

Thesis Subject: Implementation of the conceptual framework of the Internet of Things for data exchange in intelligent networks

Research object: data exchange processes in intelligent networks integrated with Internet of Things systems.

The purpose of the work: development and implementation of the Internet of Things framework for data exchange in intelligent networks, which ensures an increase in the efficiency of information exchange between IoT devices and control systems, as well as optimization of the use of system resources.

The subject of research: methods and technologies of IoT framework development to ensure effective data exchange in intelligent networks, including approaches to system design, based on microservices.

Research results

In the course of the study, a framework for integrating IoT with intelligent networks is presented, which is characterized by increased efficiency of data exchange and the possibility of scaling due to the use of microservice architecture.

Conclusion

The results of the research can be used for the implementation of IoT frameworks in smart networks for various purposes, in particular in the fields of energy, smart cities, automation of production processes and infrastructure management.

INTERNET OF THINGS (IOT), INTELLIGENT NETWORKS, DATA EXCHANGE, MICROSERVICE ARCHITECTURE, OPTIMIZATION, IOT PROTOCOLS, SYSTEMS PERFORMANCE, AUTOMATION, SMART ENVIRONMENTS.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

API - Application Programming Interface

ESB - Enterprise Service Bus

FB - Function Block

GUID - Globally Unique Identifier

HVAC - Heating Ventilation (and) Air Conditioning

IIoT - Industrial Internet of Things.

IoT - Internet of Things

IP - Intelectual Property

IPaaS - Integration Platform as a Service

JVM - Java Virtual Machine

MFB - Master Function Block

OSGi - Open Service Gateway Initiative

PIR - passive infrared sensor

PV - Photo Voltaic

PWM - pulse width modulation

QA - Quality Attribute

REST - REpresentational State Transfer

RPC - Remote Procedure Call

SAN - System Architecture and Networking

SI - International System of Units

SPF - Single Point (of) Failure

V2I - Vehicle to Infrastructure

V2V - Vehicle to Vehicle

ВСТУП

Актуальність теми.

Зростання кількості пристроїв, що підключаються до мереж, та розвиток технологій Інтернету речей (IoT) суттєво змінюють сучасний світ, впливаючи на різні галузі, такі як енергетика, транспорт, логістика, охорона здоров'я та промисловість. У зв’язку з цим виникає необхідність у забезпеченні ефективної взаємодії між цими пристроями та розробці надійних механізмів обміну даними. Технології IoT дозволяють створювати розумні середовища, такі як розумні будинки, розумні міста та промислові підприємства, що функціонують на основі збирання, обробки та аналізу великих обсягів даних. Однак інтеграція IoT з інтелектуальними мережами вимагає врахування специфічних викликів, пов'язаних з продуктивністю, масштабованістю та безпекою систем, особливо в умовах зростаючих вимог до швидкості обробки даних і низької затримки передачі інформації.

Інтелектуальні мережі, які використовують дані від IoT-пристроїв для автоматизації і оптимізації процесів, є ключовими для впровадження концепцій розумного міста та Індустрії 4.0. Вони покликані забезпечувати високу ефективність використання ресурсів, зменшення енергоспоживання та покращення якості життя. Проте для досягнення цих цілей необхідно розробити ефективні інструменти та фреймворки, що дозволяють інтегрувати численні IoT-пристрої в єдину систему управління, забезпечуючи при цьому безперервний, швидкий та безпечний обмін даними.

Актуальність теми дослідження обумовлена також тим, що наявні підходи до обміну даними в IoT-системах часто не враховують потреби сучасних інтелектуальних мереж у забезпеченні низької затримки, високої надійності та масштабованості. Традиційні архітектури систем IoT не завжди здатні задовольнити вимоги великих мережевих інфраструктур, де кількість пристроїв, типи комунікацій та обсяги переданих даних постійно зростають. Тому розробка нових фреймворків та архітектур для ефективного обміну

даними є важливим напрямком досліджень, який сприяє подоланню існуючих обмежень та забезпечує розвиток сучасних інформаційних технологій.

Крім того, у зв'язку з розвитком розподілених обчислень та застосуванням мікросервісної архітектури в IoT-системах, дослідження ефективних способів реалізації мікросервісів для моніторингу та обміну даними стає ще більш актуальним. Це дозволяє значно знизити навантаження на сервери, підвищити гнучкість системи та забезпечити можливість динамічного розподілу обчислювальних ресурсів, що є критично важливим для інфраструктур із високим навантаженням.

Отже, актуальність даного дослідження полягає в розробці ефективного фреймворку для обміну даними на основі Інтернету речей в інтелектуальних мережах, що дозволяє оптимізувати взаємодію між IoT-пристроями, підвищити швидкість передачі даних, забезпечити безпеку та адаптивність до змінних умов середовища, що в кінцевому результаті сприяє створенню більш гнучких і ефективних розумних систем.

Мета дослідження - розробка та імплементація фреймворку Інтернету речей для обміну даними в інтелектуальних мережах, що забезпечує підвищення ефективності обміну інформацією між IoT-пристроями та системами управління, а також оптимізацію використання системних ресурсів.

Об’єкт дослідження - процеси обміну даними в інтелектуальних мережах, інтегрованих із системами Інтернету речей.

Предмет дослідження - методи та технології розробки фреймворку IoT для забезпечення ефективного обміну даними в інтелектуальних мережах, включаючи підходи до проектування системи, на основі мікросервісів.

Відповідно до мети роботи було сформовано наступні **задачі:**

- Провести аналіз загальної архітектури IoT та її інтеграції з інтелектуальними мережами.
- Дослідити особливості комунікації в інтелектуальних мережах і класифікувати протоколи IoT для таких середовищ.
- Визначити функціональні та нефункціональні вимоги до фреймворку IoT для обміну даними в інтелектуальних мережах.
- Розробити концепцію застосування IoT для обміну даними в інтелектуальних мережах, враховуючи потреби сучасних технологій.
- Реалізувати та протестувати мікросервіс моніторингу та обміну даними на основі IoT для підтвердження ефективності запропонованого фреймворку.

Методи дослідження.

У роботі використані методи системного аналізу для оцінки архітектури та функціональності IoT-систем, методи моделювання для розробки концепції фреймворку, а також експериментальні методи для тестування та оцінки продуктивності реалізованого мікросервісу.

Наукова новизна отриманих результатів

У ході дослідження представлено фреймворк для інтеграції IoT із інтелектуальними мережами, що відрізняється підвищеною ефективністю обміну даними та можливістю масштабування завдяки використанню мікросервісної архітектури. Обґрунтовано переваги застосування запропонованого підходу для зменшення затримок у передачі даних і оптимізації використання ресурсів у складних розподілених системах.

Практичне значення магістерської роботи

Результати дослідження можуть бути використані для впровадження IoT-фреймворків у розумних мережах різного призначення, зокрема у сферах енергетики, розумних міст, автоматизації виробничих процесів та управління інфраструктурою. Запропоновані методи та підходи можуть бути застосовані для покращення швидкості та ефективності обміну даними між пристроями та підвищення надійності функціонування систем.

Структура магістерської роботи. Робота складається зі вступу, трьох розділів та висновків. Загальний обсяг роботи становить 83 сторінки, і містить 27 рисунків, список використаних джерел із 54 найменувань.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ФУНКЦІОНУВАННЯ ІНТЕЛЕКТУАЛЬНИХ МЕРЕЖ ТА СИСТЕМ ІНТЕРНЕТУ РЕЧЕЙ

1.1. Представлення загальної архітектури Інтернету речей

Розвиток сенсорів та технологій передачі даних відкриває можливості для нових застосувань в контексті Інтернету речей (IoT). Розумні будівлі є перспективними напрямками для таких застосувань, де прилади можуть регулюватися відповідно до зайнятості будівлі для досягнення енергозбереження, підвищення комфорту та реалізації нових послуг. У цьому проєкті досліджується універсальна платформа для розробки застосувань, збору та адміністрування даних. Ця платформа призначена для надання структурованої інфраструктури, інструментів та функціональних перевірок для побудови застосувань на її основі. Використовуваний кейс – це інтерактивна панель розумної електромережі, яка включає пристрої, такі як розумні лічильники, фотоелектричні (PV) панелі, системи клімат-контролю та інші електроприлади, що генерують дані та можуть керуватися дистанційно. Ця інформація збирається з самих пристроїв або з навколишнього середовища та додатково агрегується з сторонньою інформацією. В результаті реалізації цієї платформи нові застосування можуть розроблятися та розгортатися швидше, є більш адаптованими до доступної інфраструктури та можуть повторно використовуватися. Застосування, такі як інтелектуальне освітлення або системи клімат-контролю, можуть бути розроблені для підвищення комфорту людей або зменшення споживання енергії.

Сучасні програмні платформи є надто абстрактними, накладають обмеження на топологію або мають велике та статичне навантаження. Вони вимагають глибоких знань для розробки, розгортання та переробки

застосувань, що вимагає розуміння роботи всієї системи без високого рівня огляду на взаємодіючі компоненти.

Пропонована платформа намагається подолати ці обмеження. Зацікавлені сторони мають різні інтереси та погляди на те, як ці застосунки повинні функціонувати. Водночас відсутні чіткі рекомендації щодо розробки застосувань.

За допомогою простих функціональних блоків (FB) підтримується легкий для розуміння огляд дизайну, а також підвищується повторне використання та модульність.

Розвиток технологій пристроїв призвів до збільшення їх функціональності та здатності генерувати дані. Удосконалення обчислювальних можливостей, а також зберігання, зниження ціни та підключення в апаратному забезпеченні має перевагу в тому, що (малі) підприємства або окремі особи можуть дозволити собі впровадження підключеної мережі речей [9]. За допомогою цих мереж створюються різні додатки для нових або вдосконалених послуг. Наприклад, сітка датчиків температури та вологості в теплиці дає власнику важливу інформацію про те, які дії потрібно вжити, щоб підвищити продуктивність і якість його врожаю.

Крім того, щомісяця підключається до 328 мільйонів пристроїв, згідно з [25], із загальним прогнозом понад 170 мільярдів до 2025 року. Таким чином, загальна кількість створених даних також збільшується, що породжує необхідність грамотного управління інформацією у масштабований спосіб. Це робиться шляхом впровадження локальної обробки або фільтрації даних. Коли обчислювальні можливості перевищуються, дані, як правило, передаються в хмару, де системи з більшою потужністю та алгоритми аналізу можуть їх обробити. Крім того, хмарні обчислення мають перевагу кластеризації даних із кількох джерел , отже, маючи можливості знаходити кореляції та отримувати більше інтелектуальних даних із доступних наборів даних.

Існує багато додатків, які можна розробити для підвищення комфорту, впровадження нових бізнес-кейсів або підвищення ефективності та зменшення витрат у різних галузях. Однак для програм потрібні дані з пристроїв або навіть контроль над ними, а також вказівки щодо того, як їх можна створити. Це основний стимул для впровадження структури для розміщення та керування цією взаємодією.

У той же час прогрес в алгоритмах машинного навчання відкрив можливість співвідносити дефіцитні дані з уявленнями людей щодо певних подій. Наявність стандартизованої системи обміну даними ще більше сприятиме використанню цих технологій.

Ця концепція збору даних, спільного використання, агрегації та активації середовища узагальнено під назвою Інтернет речей (IoT). Основні компоненти архітектури IoT показано на рисунку 1.1.

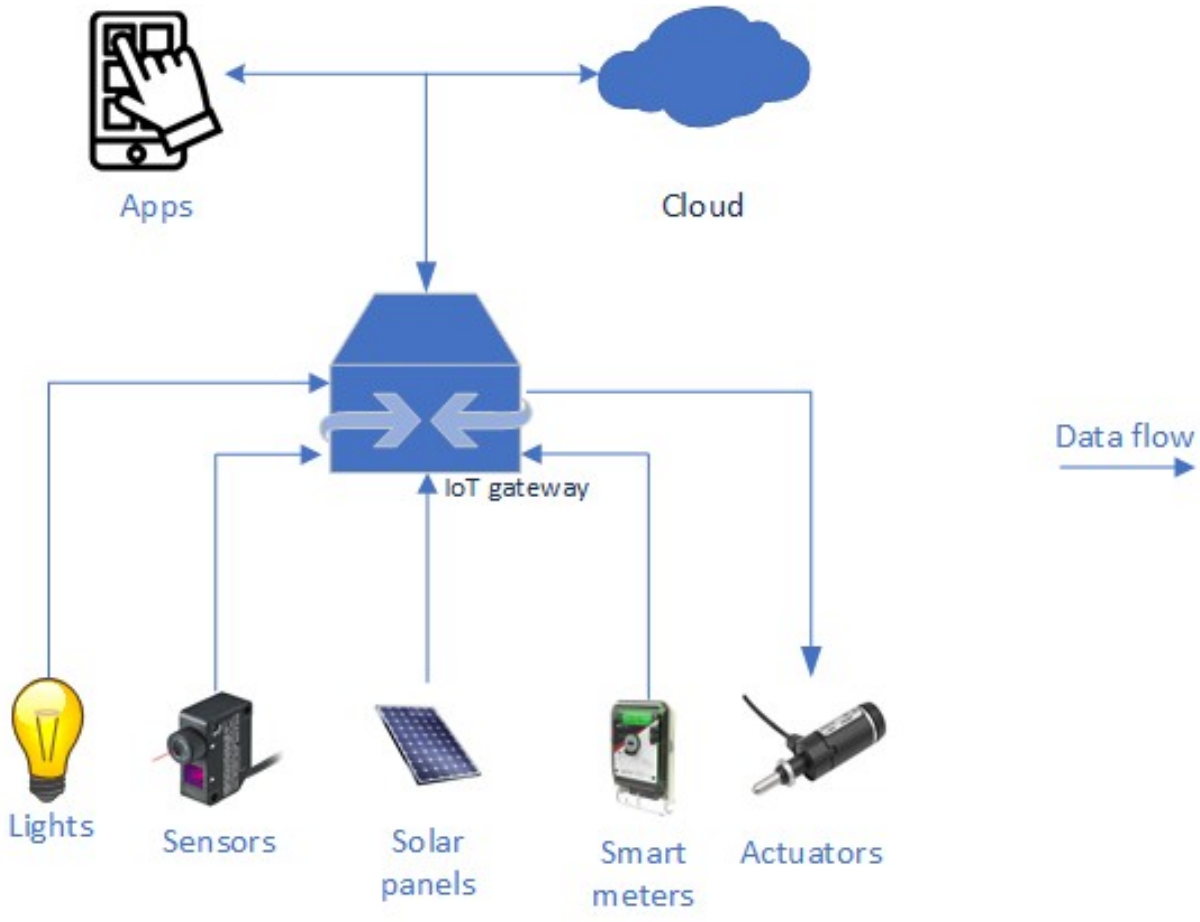

Рис. 1.1. Загальна архітектура IoT

Різні пристрої підключаються до шлюзу, який, у свою чергу, підключається до хмарних служб, а також програм (наприклад, інформаційних панелей користувачів). Сам шлюз може бути представлений сервером або сукупністю підсистем, відповідальних за розміщення таких послуг, як зберігання даних або керування пристроєм/додатком.

1.2. Опис області дослідження

Загалом програми IoT реалізуються навколо центрального вузла обробки [4 -6]. Це означає, що кілька джерел даних націлені на один пристрій, який, у свою чергу, агрегує цю інформацію та повертає її у формі послуг.

З точки зору розробки, це зручна інфраструктура для доставки нових додатків, оскільки доступні функції можна використовувати повторно. Однак це також означає, що нові програми повинні підкорятися обмеженням, наданим можливостями центрального вузла. Ці обмеження можуть стосуватися використання певних протоколів зв'язку / обмежених ресурсів на центральному вузлі або навіть використання лише певного асортименту готових продуктів.

Завдяки централізованій архітектурі зі збільшенням кількості підключених пристроїв і функціональних можливостей система стає роздутою, що впливає на продуктивність служб .

Крім того, з точки зору функціональності, незважаючи на те, що розробка додатків є ітеративною, вона все ще виконується статично. Це означає, що зацікавлені сторони опитуються, а розробники додатків відображають їхні потреби через вимоги до системи. Прозорість розробки не використовується далі, а відсутність стандартизації в цьому процесі впливає на можливість повторного використання та життєві цикли програм. Підтримувати таку постійно зростаючу структуру стає складно, а передавати

знання іншим розробникам, які не знайомі з базовою інфраструктурою, дещо складно.

Розробка додатків і пристроїв не обмежувалася, щоб забезпечити повні рішення. Це означає, що розробники додатків зосереджені на розробці алгоритмів без урахування апаратних обмежень. З іншого боку рішення, розробники апаратного забезпечення також зосереджені на питаннях, близьких до їх власної сфери знань, таких як забезпечення конкурентоспроможних продуктів з точки зору продуктивності пристрою, витрат виробництва та інноваційних функцій. З точки зору розгортання, розробники додатків припускають, що пристрої створюють суворі набори даних для реалізації своїх алгоритмів, тоді як розробники апаратного забезпечення стурбовані лише створенням надійних та інноваційних продуктів, здатних взаємодіяти з логікою вищого рівня та надавати інформацію на місці.

У той же час, залучення інших зацікавлених сторін, окрім розробників, ускладнюється через відсутність прозорості щодо того, як ці рішення розробляються та як вони фактично функціонують. Також немає чітких вказівок щодо того, як слід розробляти ці рішення, і, як наслідок, існує багато проблем із розробкою масштабованого, надійного та багаторазового програмного забезпечення.

Хоча занепокоєння окремих сторін не завжди цікаві для інших зацікавлених сторін, вони важливі для належного функціонування рішення, і наразі цей аспект не розглядається.

1.2.1. Очікувана ситуація

Розробникам не слід турбуватися про вимоги, які виходять за межі їх сфери знань. І розробники додатків, і розробники апаратного забезпечення повинні турбуватися лише про продуктивність або функціональність своєї роботи, а не про те, як її обробляють інші зацікавлені сторони в кінцевому

продукті. Наприклад, розробників додатків не цікавить протокол зв'язку, який використовується для взаємодії з пристроями з метою отримання даних. З точки зору керування пристроями, розробники додатків хочуть контролювати параметри апаратного забезпечення, але їх не цікавить, як пристрої насправді цього досягають. Як приклад, певний кут серводвигуна може мати особливе значення для застосування, але точний робочий цикл сигналу широтно-імпульсної модуляції, який використовується локальним контролером для досягнення цього заданого значення, виходить за рамки вищих управління.

З іншого боку, розробникам апаратного забезпечення не слід турбуватися про надання своїх даних у певному форматі або враховувати показники продуктивності, які не впливають на якість їх продукту. Прикладом цього може бути обсяг пам'яті, який може знадобитися системі для зберігання всіх даних із датчика, або той факт, що ця інформація надається в іншій Міжнародній системі одиниць (СІ), ніж вона використовується в програмі.

Бажано, щоб різні зацікавлені сторони могли налаштовувати свої програми залежно від своєї мети. Наприклад, кінцеві користувачі додатків зацікавлені в надійній та чутливій системі для взаємодії, тоді як мережевий адміністратор буде зацікавлений у надійній інфраструктурі. Це може призвести до наявності подібної програми, яка потребує менших витрат на зв'язок. Таким чином, поточну ситуацію, коли занепокоєння зацікавлених сторін відображено за допомогою системних вимог, можна додатково посилити за допомогою налаштованих обмежень, таких як затримка, накладні витрати на зв'язок, ємність зберігання або обчислювальна потужність.

Деякі з цих обмежень і цілей можуть бути спільними або навіть конфліктними для різних сторін, тому для забезпечення сумісності системи необхідні визначення меж.

1.2.2. Мета проекту

Структура проміжного програмного забезпечення сприяла б розробці, підтримці та внеску більшої кількості зацікавлених сторін у ці програми. Сфера застосування цієї структури полягатиме в інтерфейсі як з пристроями, так і з додатками, для підвищення прозорості, надійності та можливості повторного використання, для вирішення окремих проблем зацікавлених сторін і надання вказівок для розробки таких рішень.

Структура повинна відповідати вимогам розробників пристроїв і додатків, а також інформувати їх у разі суперечливих елементів, по суті, діючи як арбітр. Наприклад, даний датчик може мати регульовану вихідну швидкість передачі даних. Однак, коли налаштування програми перевищують ці межі, апаратне забезпечення має застосувати найближчі можливі налаштування, і в той же час структура повинна зробити прозорим, щоб бажані та фактичні налаштування не узгоджувалися.

Існує кілька фреймворків, метою яких є сприяння реалізації програм. Більш широкий перелік цих структур представлено в [21]. Кожен з цих фрейморків має свої переваги та обмеження. Деякі відповідні фреймворки далі аналізуються в роботі разом із концепціями, які визначає сервіс-орієнтована архітектура (SOA) і які підходять для домену IoT.

Простота розробки та розгортання додатків є основними критеріями вибору фреймворку, але це досить суб’єктивний підхід. Модульність, накладні витрати на зв’язок, збій єдиної точки (SPF), затримка або пропускна спроможність даних – це показники, за якими можна розрізнити фреймворки, але відповідні ключові показники ефективності кожного поля IoT можна визначити на замовлення.

Вищезазначені показники є актуальними для теми інтелектуальної мережі, і вони вмотивовані в аналізі (літературних) сучасного стану поточних доступних структур.

1.3. Особливості комунікації в розумних мережах

Успіх взаємодії в реальному часі між елементами енергосистеми, як основне завдання розумної мережі, залежить від впровадження безпечних, надійних, масштабованих, інтегрованих, взаємодіючих та універсальних систем комунікації. Рисунок 1.2 демонструє комунікаційну мережу в розумній мережі, яка класифікується на три домени, включаючи Домашню мережу (HAN), Польову мережу (FAN) та Широкомовну мережу (WAN). HAN охоплює споживацьку сторону, включаючи побутові прилади, розподілені джерела енергії (DER) та електромобілі (EV), і вимагає таких застосувань, як Система управління енергією в домогосподарстві (HEMS), Vehicle-to-Grid (V2G) та інвертори.

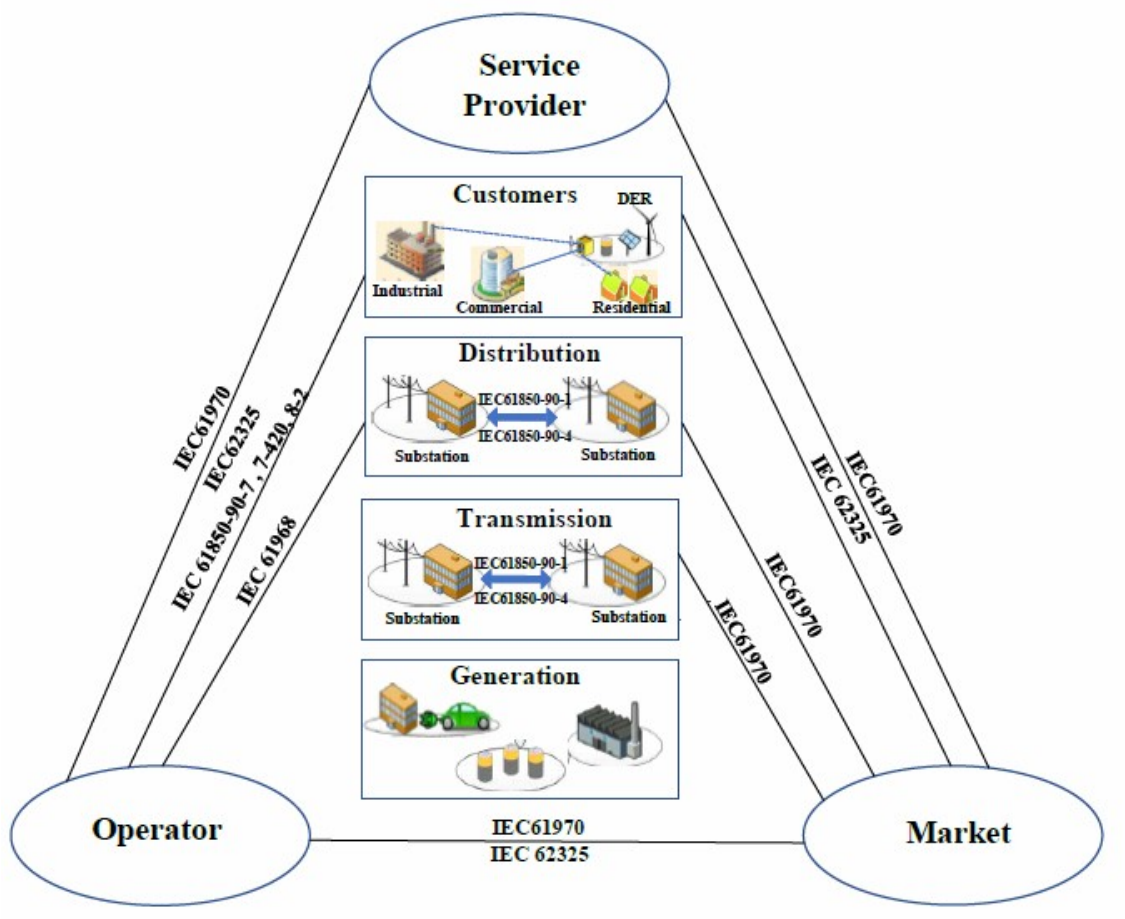

Рис. 1.2. Концептуальна модель розумної мережі та її відповідні стандарти

FAN - це домен, який організовує взаємозв'язок між стороною споживача та електромережею. На цьому рівні спілкування концентратори збирають дані від лічильників споживачів та DER для наглядового рівня. Оператори енергосистем застосовують цю інформацію для надання послуг у WAN, таких як DR, система управління розподілом (DMS) та широка ситуаційна обізнаність (WASA).

Крім того, AMI може бути у всіх трьох доменах на основі політики комунальних послуг. Оскільки він збирає інформацію від вимірювальних приладів на стороні споживання до наглядового боку, ми вважаємо AMI частиною домену FAN. Оскільки основною метою цієї статті є дослідження відповідного проміжного програмного забезпечення для застосування розумної мережі, попередня інформація необхідна для визнання комунікаційних вимог кожного домену.

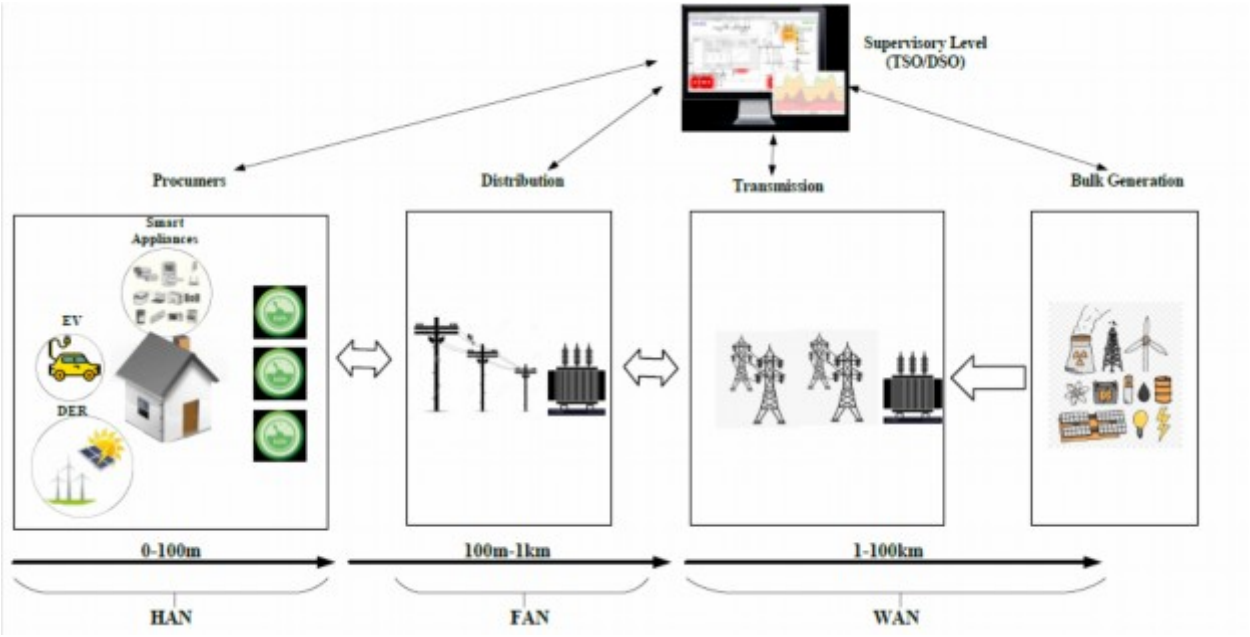

Рис. 1.3. Рівень зв'язку в розумній мережі

Основні характеристики системи зв'язку в розумній мережі: безпека, надійність та захищеність обміну даними, що дозволяє впровадження кількох стандартів. Таблиця 3 узагальнює основні стандарти з великої кількості стандартів IEC та кількох стандартів IEEE, які підтримують розумну мережу.

Передісторія серії стандартів IEC 61850 включає поширення використання IED в енергосистемі, що вимагає зв'язку в реальному часі для забезпечення управління, моніторингу, вимірювання та захисту на підстанції. Компоненти SAS сприяють моніторингу, управлінню та конфігурації підстанції на трьох рівнях:

- рівні процесу,
- рівні секції,
- рівні станції.

Датчики та виконавчі механізми розташовані на шині процесу на рівні секції та надсилають інформацію, таку як вимірювання струму та напруги, до IED. IED виконують управління, моніторинг та захист шляхом обробки інформації, отриманої від рівня процесу. Рівень станції є місцем розташування нагляду за енергомережею, включаючи бази даних, операторів та інтерфейси для дистанційного керування та зв'язку. Цей стандарт визначає три типи повідомлень для взаємодії з цими рівнями зв'язку. MMS використовується для некритичної інформації з низьким або середнім пріоритетом та у форматі запит-відповідь. З іншого боку, критична інформація з високим пріоритетом, така як сигнали спрацювання, використовує службу Generic Object Oriented Substation Event (GOOSE), а вимірювальні одиниці з високим пріоритетом використовують службу Sampled Value (SV). Хоча GOOSE і SV мають багатоадресний формат, кожне з цих повідомлень має обмеження часу та відображається відповідно до стеку зв'язку, запропонованого в частині 8–1 стандарту, який використовує модель відкритої системи взаємозв'язку (OSI) та Ethernet як фізичний рівень у середовищі локальної мережі.

Останнє видання стандарту було розглянуто як швидка інновація технології та розділило частину моделі даних та модель зв'язку для вирішення цього питання. Цей стандарт також розширив інформаційну модель для підтримки розподілених джерел енергії в частині 7–420, 90–7 та надав стек зв'язку для взаємозв'язку в глобальній мережі на основі

застосування XMPP для відображення інформації в частині 8–2. Такі розробки роблять цей стандарт відповідним для використання в середовищі зв'язку розумної мережі.

Однак інша література та лабораторний досвід пропонують інші протоколи IoT, а не XMPP, оскільки їх продуктивність була кращою в забезпеченні якості обслуговування, вимог до інфраструктури реалізації та специфікацій майбутнього розвитку.

1.4. Класифікація протоколів IoT на основі застосування в розумній мережі

1.4.1. Архітектура та специфікація протоколів IoT

Основною проблемою реалізації розумної мережі є комунікація різнорідних розподілених елементів. Програмне забезпечення посередництва функціонує як інтерфейс послуг та програмних додатків у комунікаційній архітектурі для полегшення цієї взаємодії шляхом приховування складності операційної системи для розробників програмного забезпечення додатків. Багато хто оцінює, що програмне забезпечення посередництва прискорює інтеграцію різнорідних сутностей, збір інформації, безпеку обміну даними та оцінку ситуації в розумній мережі Опис та характеристики більш широко розгорнутих протоколів IoT в розумній мережі, які відображаються на IEC 61850 або CIM, є наступними.

AMQP є відкритою стандартною архітектурою протоколу публікація-підписка, представленою в 2003 році та стандартизованою пізніше Організацією сприяння стандартизації структурованої інформації (OASIS) у 2011 році. Зображені на рисунку 1.4, Exchange, Queue та Binding є трьома елементами, які встановлюють передачу повідомлень у цьому протоколі. Exchange, який є брокером цього протоколу, надсилає повідомлення до черги на основі їх пріоритетів. Binding визначає пріоритет різними методами, включаючи direct, topic та fanout. Transport Layer Security (TLS) пропонує

безпеку передачі в цьому протоколі. AMQP не є легковажним протоколом і, у випадку пам'яті, пропускної здатності та потужності, він не може підтримувати автономні датчики.

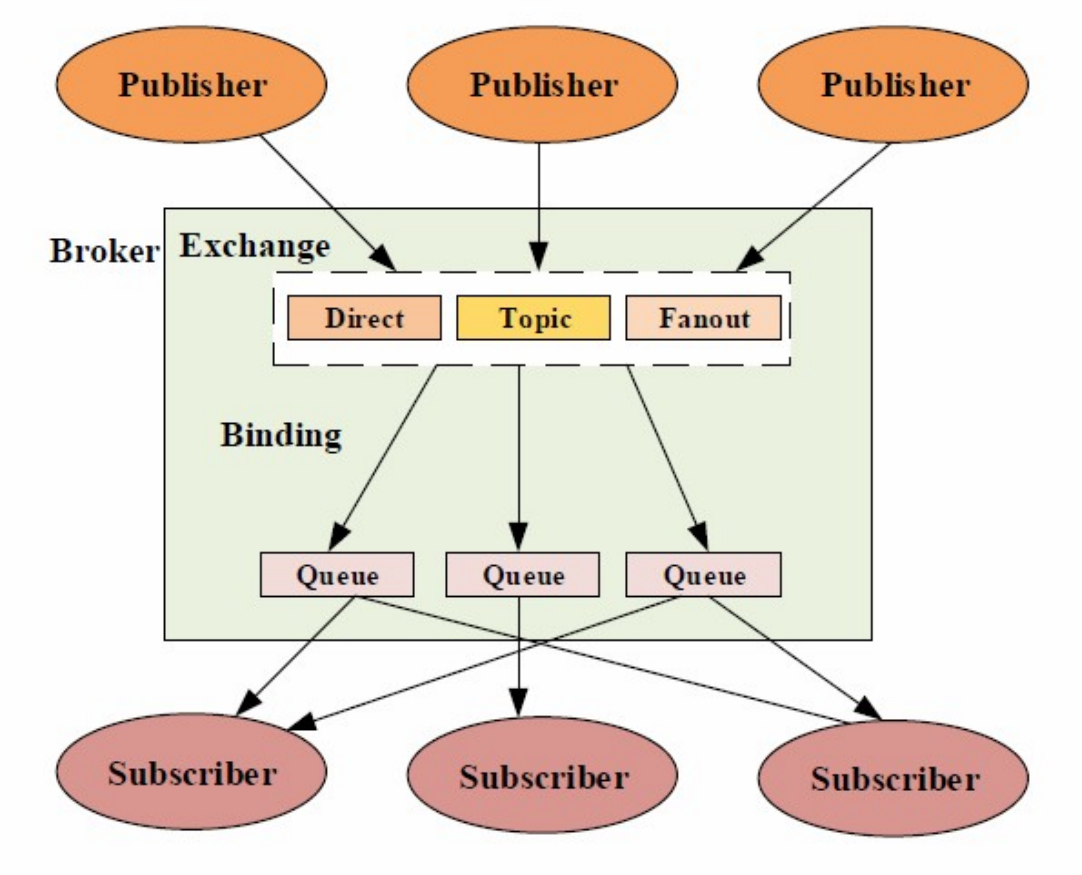

Рис. 1.4. Архітектура черги повідомлень Advanced Message Queue Telemetry Transport (AMQP)

CORBA є програмним забезпеченням посередництва, визначеним Групою управління об'єктами (OMG), для полегшення спеціалізованої мови та незалежної від платформи взаємодії розподілених об'єктів. Об'єкти в цьому протоколі можуть бути клієнтами або серверами, які спілкуються через брокер запитів об'єктів (ORB).

Відповідно до рисунка 1.5, ORB, який є ядром моделі CORBA, має кілька інтерфейсів, визначених у мові визначення інтерфейсів (IDL) або розташованих у службі репозиторію інтерфейсів. Хоча клієнт надсилає запит за допомогою інтерфейсу динамічного виклику (DII), заглушок IDL або інтерфейсів ORB, сервер отримує запити через інтерфейс динамічного

скелета (DSI) та скелет IDL. Адаптер об'єкта, інший інтерфейс між ORB та сервером, відповідає за відображення посилань на об'єкт у ORB на відповідний об'єкт на сервері.

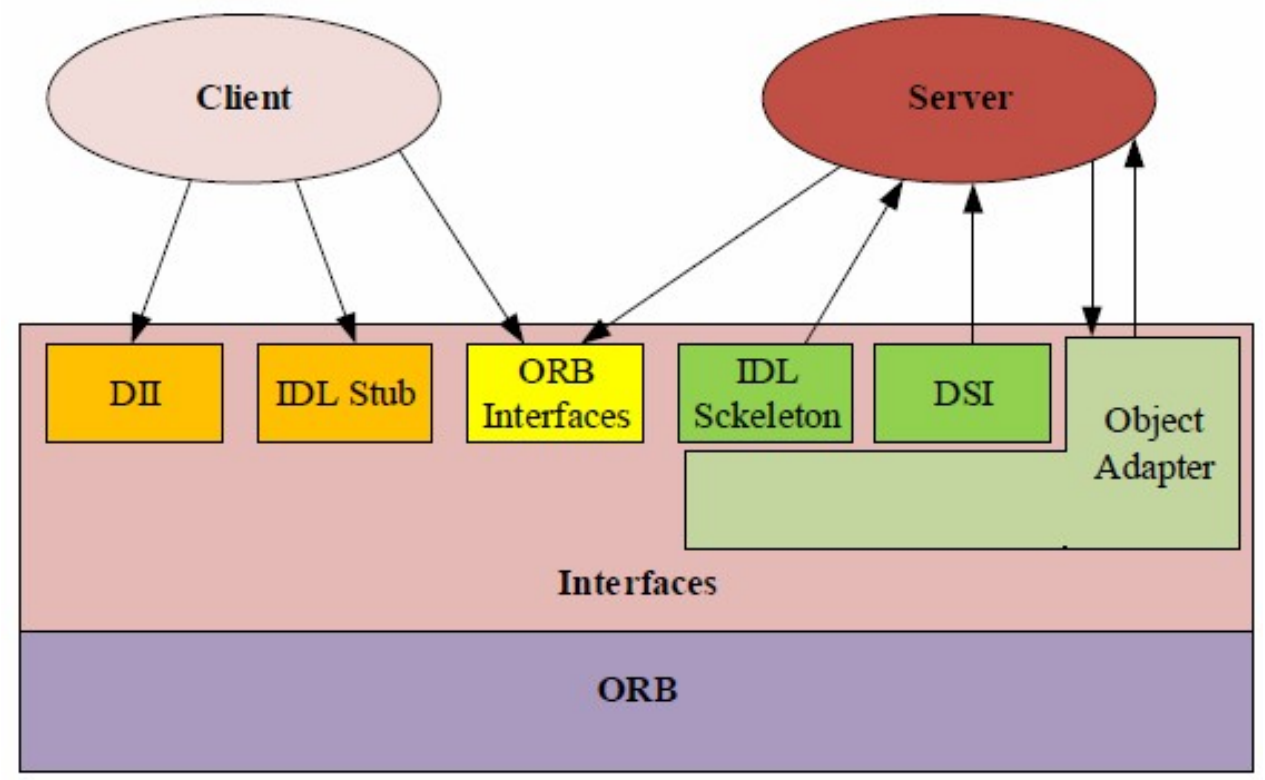

Рис. 1.5. Загальна архітектура посередника запитів об’єктів (Common Object Request Broker Architecture CORDA)

DDS (Data Distribution Services) є відкритим стандартним програмним забезпеченням посередництва, розробленим OMG, яке забезпечує зв'язок у реальному часі через модель повідомлень публікація-підписка. DDS отримує перевагу від того, що учасникам більше не потрібно знати один одного, застосовуючи методи виявлення, включаючи метод виявлення Data Centric Publisher Subscriber (DCPS) або Real-Time Publisher Subscriber (RTPS). Тому DDS є безброкерним протоколом обміну інформацією без ризику збоїв вузького місця.

Відповідно до концепції DCPS, представленої на рисунку 1.6, існує доменний простір, у якому всі додатки можуть взаємодіяти через нього, і всі комунікаційні сутності розміщені в домені. Учасники домену включають

Data Reader, DataWriter, Publisher та Subscriber. Учасники мають доступ до даних на основі доменної теми та типу.

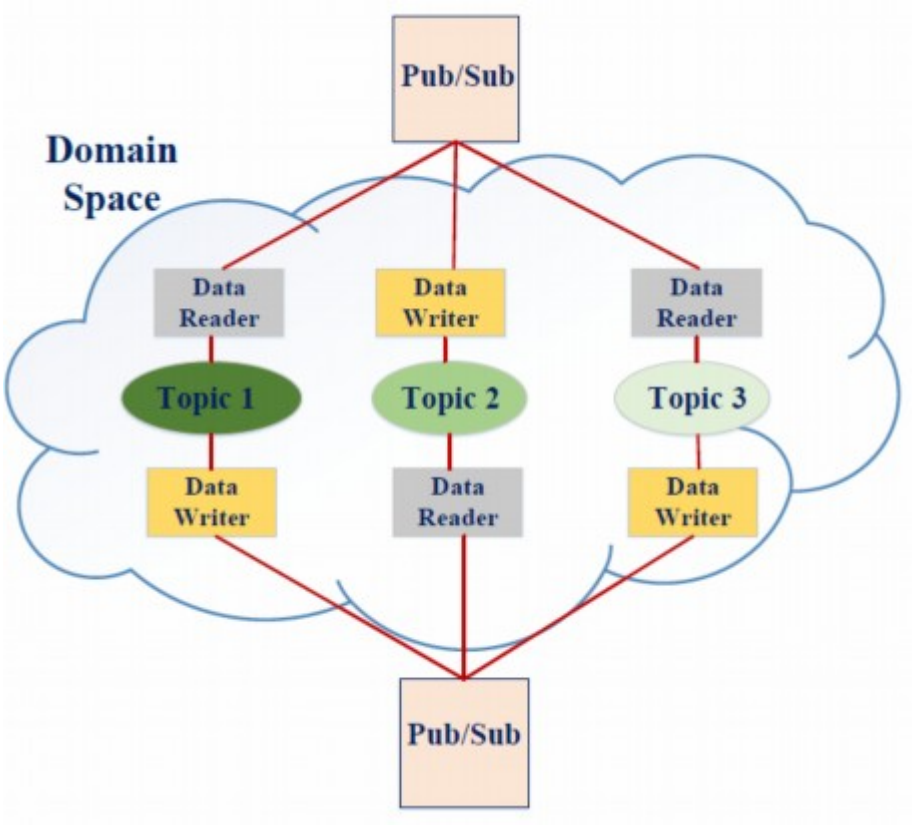

Рис. 1.6. Доменний простір служб розподілу даних

OPC (Open Platform Communications United Architecture) є результатом співпраці галузі автоматизації у наданні інформації пристрою для застосування без доступу до моделі пристрою та продуктивності на основі моделі об'єкта компонента Microsoft у 1994 році. Метою OPC є підтримка взаємодії з іншими операційними системами за допомогою єдиної архітектури, що називається OPC UA. Цей протокол був стандартизований IEC 62541 з архітектурою клієнт-сервер. Як показано на рисунку 1.7, OPC UA має два хребти в своїй архітектурі, тобто модель транспорту та модель даних.

Хоча сервери спілкуються з клієнтами за допомогою двох різних типів транспортування, включаючи двійковий, який називається UA native, або Simple Object Access Protocol (SOAP)/HTTP, який викликає веб-служби UV

через TCP/IP, модель даних визначає правила для серверів щодо того, як зображати об'єкти, включаючи змінні та методи, через адресний простір для клієнтів. Базовий рівень служби надає послуги в обмін на інформацію. Ця інформаційна модель протоколу, включаючи Data Access (DA), Alarms and Conditions (AC), Historical data Access (HA) та Programs (PRG), може бути прийнята з інформаційними моделями інших організацій, таких як IEC та Міжнародне товариство автоматизації (ISA), або з інформаційними моделями постачальників.

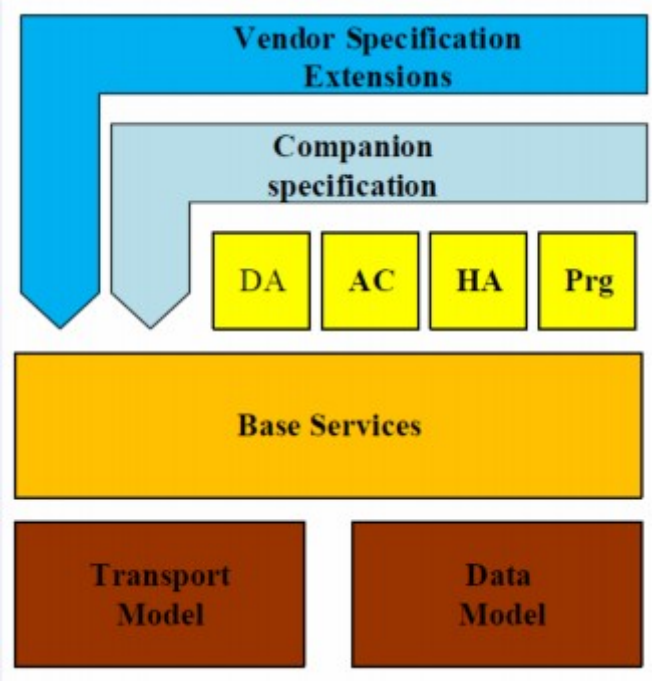

Рис. 1.7. Об'єднана архітектура комунікацій відкритої платформи

Архітектура безпеки OPC UA готує автентифікацію, авторизацію, конфіденційність, цілісність, аудитованість, доступність, транспорт, зв'язок та рівні застосування. Хоча TLS забезпечує шифрування для протоколу HTTP на транспортному рівні, рівень зв'язку забезпечує конфіденційні повідомлення та цілісність. Рівень застосування містить автентифікацію та авторизацію користувача в сеансі. OPC UA також підтримує модель публікація-підписка для застосування меншої транзакції сеансу за допомогою пакетів UDP. Ця архітектура є більш єдиною, ніж інші, тому її

можна застосовувати для всіх типів застосунків IoT, оскільки вона охоплює необхідні функції застосунків, розроблених у галузі IoT.

1.4.2. Перспективи використання IoT і розумних мереж

Застосовуючи MAS (Multi-Agent System) та багатомікромережу в розумній мережі, зростає кількість активних вузлів в енергосистемі, що беруть участь в AS. Цей намір призводить до високого проникнення датчиків, виконавчих механізмів та обчислювальних одиниць у концепції IoT.

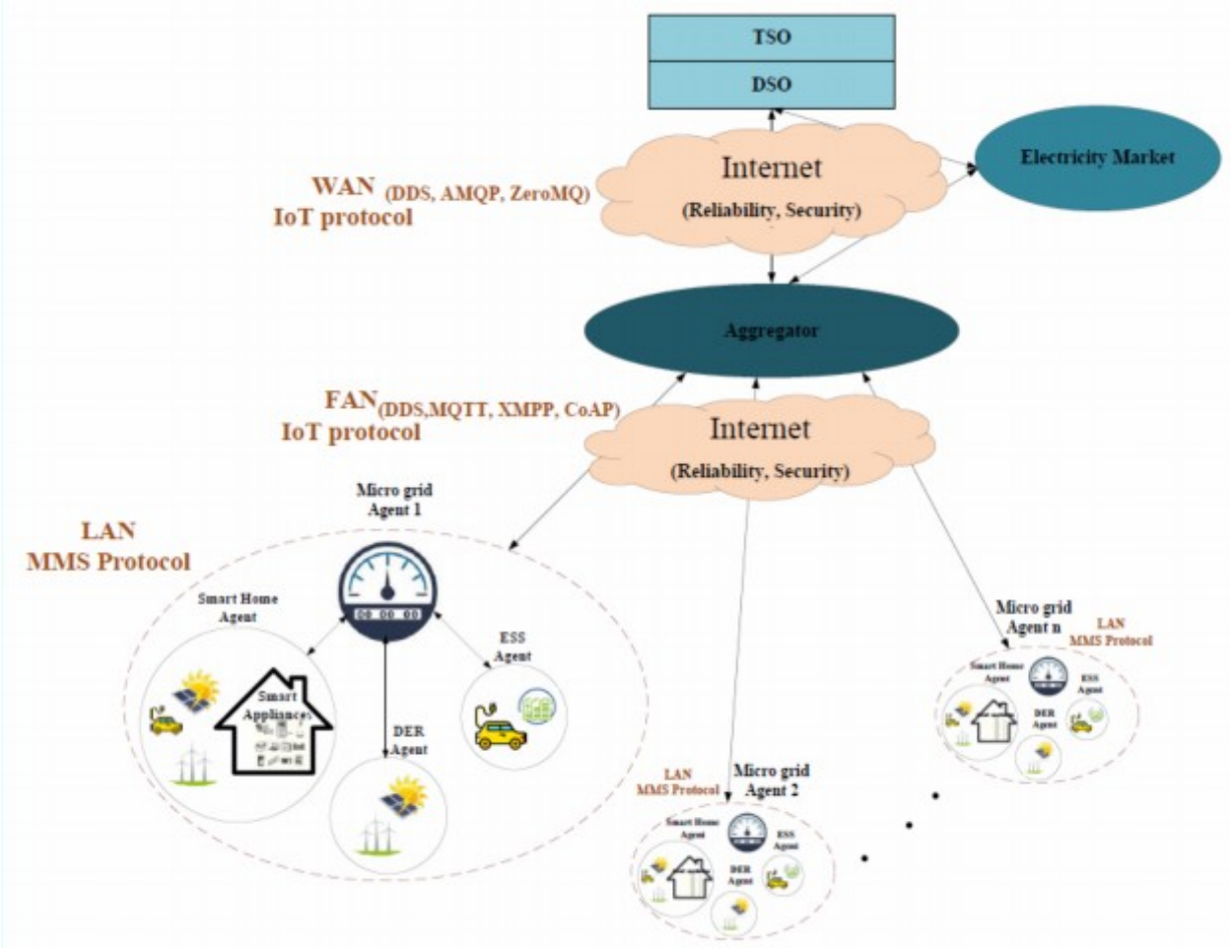

Рис. 1.8. Архітектура розумної мережі на основі протоколу IoT

Хмарні обчислення та туманні обчислення є інструментами для полегшення цього сценарію. Оскільки не існує спеціалізованого стандарту для їх застосування в середовищі розумної мережі, реалізація комунікаційної

структури, стандартів та протоколів цього явища заохочує майбутні дослідження в цій галузі. Кінцеві користувачі збільшують свій внесок у AS енергомережі та використовують Інтернет як інфраструктуру зв'язку для випуску спеціалізованих, які є вразливими. У цьому випадку конфіденційності ця перевага визначає новий горизонт цілісності даних, автентифікації, конфіденційності, конфіденційності даних, стандарту, пропускної здатності та вимог до затримки протоколу IoT у дослідженнях розумної мережі.

Висновки до розділу

У цьому розділі було проведено дослідження архітектури Інтернету речей (IoT) та особливостей функціонування інтелектуальних мереж. Визначено основні компоненти IoT, їхню взаємодію та ключові протоколи, які забезпечують ефективну комунікацію між пристроями в розумних мережах. Проаналізовано різні протоколи IoT з точки зору їх застосування в розумних мережах та розглянуто перспективи їхнього розвитку. Особлива увага приділена аналізу архітектури IoT та специфікацій протоколів, які підтримують надійний обмін даними в умовах інтелектуальних мереж.

РОЗДІЛ 2. МЕТОДОЛОГІЧНІ ОСНОВИ ЗАСТОСУВАННЯ ТЕХНОЛОГІЙ ІНТЕРНЕТУ РЕЧЕЙ В КОНТЕКСТІ ІНТЕЛЕКТУАЛЬНИХ МЕРЕЖ

2.1. Дослідження концепцій Інтернету речей

Інтернет речей — це технологічна революція, яка представляє майбутнє обчислювальної техніки та комунікацій, і його розвиток залежить від динамічних технічних інновацій у низці важливих галузей, від бездротових датчиків до нанотехнологій. Інтернет речей (IoT) — це структура, в якій усі речі представлені та присутні в Інтернеті. Зокрема, Інтернет речей націлений на пропонування нових додатків і послуг, що об'єднують фізичний і віртуальний світи, у яких зв'язок між машинами (M2M) є базовим зв'язком, що забезпечує взаємодію між речами та додатками в хмарі.

Одна з головних проблем IoT полягає в тому, що це настільки велике й широке поняття, що не існує запропонованої єдиної архітектури. Для того, щоб ідея IoT працювала, вона повинна складатися з асортименту сенсорних, мережевих, комунікаційних і обчислювальних технологій, серед іншого.

Згідно з рекомендаціями Міжнародного союзу електрозв'язку (ITU), мережева архітектура Інтернету речей складається з:

a) Сенсорний рівень

b) Рівень доступу

c) Мережевий рівень

d) Рівень проміжного програмного забезпечення

e) Рівні програми

Вони схожі на еталонну модель взаємозв'язку відкритих систем (OSI) у мережі та передачі даних.

Базова архітектура Інтернету речей в основному поділяється на 3 типи, включаючи програми, процесори та передачу даних.

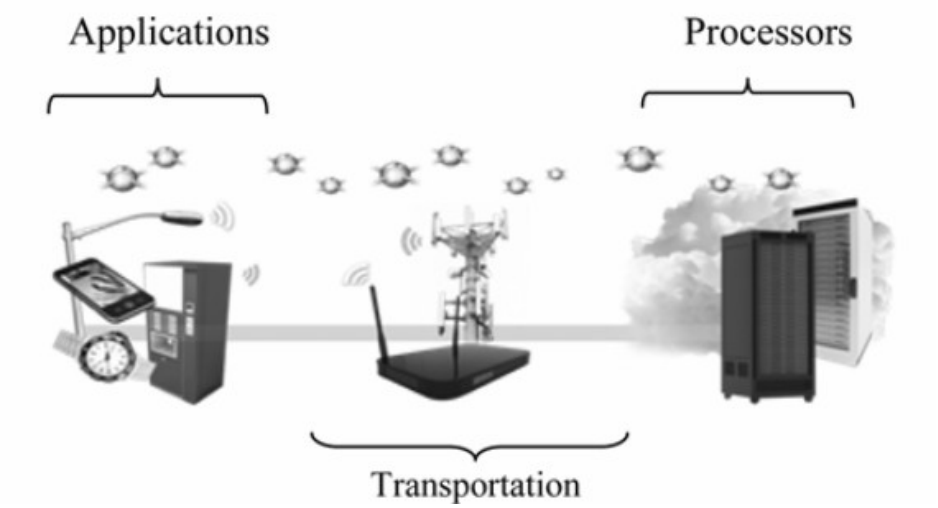

Рис. 2.1. Базова архітектура IoT

П’ятирівнева архітектура призначена для визначення повної концепції її функціонування та розвитку пристроїв IoT. Нова структура включає 5 шарів, як показано на рисунку 2.2.

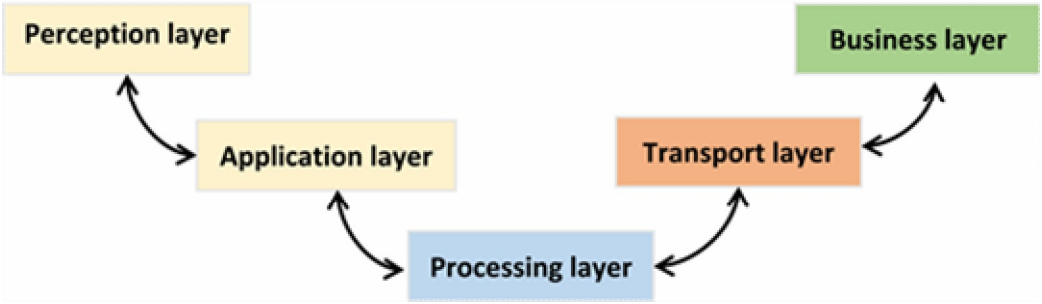

Рис. 2.2. П’яти рівнева архітектура

Рівні сприйняття та застосування функціонують так само, як і в попередній архітектурі. Основним завданням рівня обробки є обробка інформації, отриманої з мережевого рівня, і прийняття рішень на основі результатів, отриманих від повсюдних обчислень. Транспортний рівень, який передає дані датчика з рівня сприйняття на рівень обробки і навпаки через такі мережі, як бездротові мережі, LAN, 3G, LTE, RFID і Bluetooth. Нарешті, бізнес-рівень візуалізує інформацію та статистику з прикладного рівня та використовує ці знання для планування майбутніх цілей і стратегій.

Модель обміну даними на сервері відноситься до архітектури зв'язку, яка дозволяє користувачам експортувати та аналізувати дані смарт-об'єктів із хмарної служби в поєднанні з даними з інших джерел. Потім дані завантажуються до двох різних постачальників послуг додатків. Архітектура також допомагає зі збором і аналізом даних. Наприклад, промисловець зацікавлений в аналізі енергоспоживання заводу шляхом збору даних, отриманих датчиками IoT і комунальними системами.

Внутрішня модель обміну даними припускає, що уніфіковані хмарні сервіси або інтерфейси прикладних програмістів (API) підходу до хмари необхідні для досягнення інтелектуальної сумісності даних пристроїв, розміщених у хмарі. На рисунку 2.3 представлена ця модель.

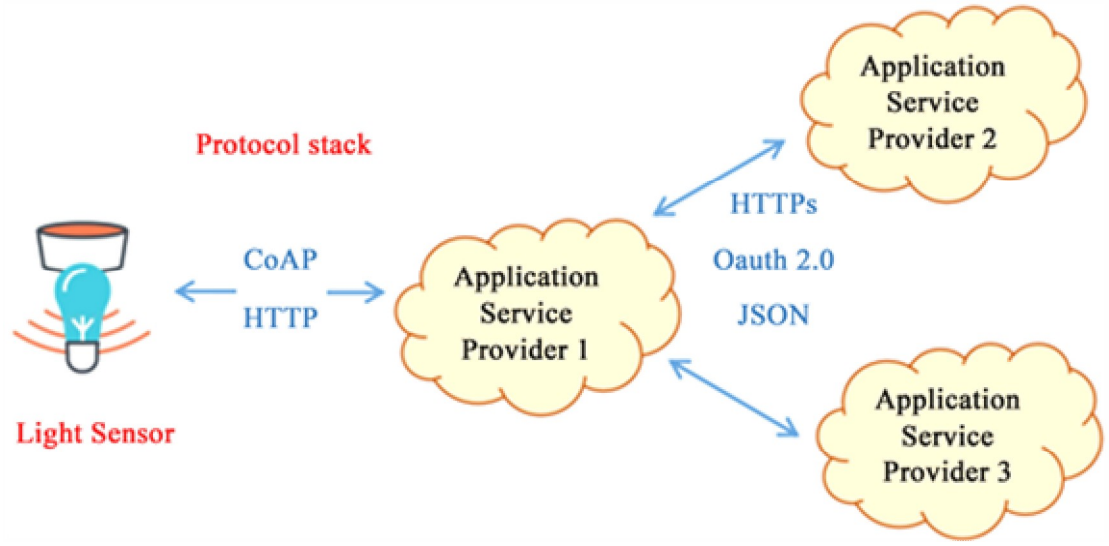

Рис. 2.3. Внутрішня модель обміну даними

Концепція IoT узагальнюється як мережа пристроїв, здатних взаємодіяти або обмінюватися інформацією з середовищем або самими собою [2]. Спільною рисою незалежно від сфери IoT є постійно зростаюча кількість пристроїв, які підключаються до мережі.

Ця мережа не обмежується з'єднанням пристроїв разом або з Інтернетом та обміном інформацією датчиків. Розширені телеметричні дані дозволяють реалізувати так звані цифрові близнюки. Маючи детальне

уявлення про стан системи, можна отримати її модель, яка може допомогти в розробці нових функціональних можливостей і надати інформацію для моделювання або навіть (прогнозного) обслуговування [3].

У наступних підрозділах описано кілька полів та їхні особливості, серед яких розумна будівля та розумна мережа, які містять фокус використання цього проекту. Також представлено їхню приналежність до ширшої теми IoT.

2.2. Розумна мобільність та Індустрія 4.0

2.2.1. Концепція розумної мобільності

Сфера розумної мобільності включає в себе всі активи, пов’язані з транспортом. Connected Car, Vehicle to Vehicle (V2V) або Vehicle to Infrastructure (V2I) є частиною цього домену, де вбудовані пристрої обмінюються даними між собою, а також з Інтернетом, щоб надавати рішення для більш ефективного та комфортного перевезення людей та піти одн.

Особливості цього домену пов’язані з необхідністю зв’язку в реальному часі та адміністрування мережі ad-hoc. Наприклад, транспортні засоби повинні мати можливість спілкуватися одна з одною для обміну даними щодо їх поточної швидкості та траєкторії руху взводу.

З подібної точки зору, сама інфраструктура (дороги, світлофори/покажчики) може мати можливості підключення та також повинна бути здатна адаптуватися до швидкого трафіку для обміну інформацією. Це передбачає встановлення швидкого, але також надійного та безпечного з’єднання між об’єктами за короткий проміжок часу.

Домен Connected Car передбачає збір інформації про транспортний засіб на місці, яка потім фільтрується, передається та/або агрегується для реалізації таких послуг, як моніторинг автопарку в реальному часі або віддалена діагностика [18 - 20]. Кількість пристроїв або активів, які

генерують дані, відносно постійна, оскільки джерелами даних є самі компоненти транспортного засобу (наприклад, двигун, трансмісія, інформаційно-розважальна система), однак обсяг даних у сучасних транспортних мережах є головною проблемою для розробки базової структури IoT для цього.

Концепція розумної мобільності, що передбачає створення інтелектуальних транспортних систем, значно трансформується завдяки технологіям V2V (Vehicle to Vehicle) та V2I (Vehicle to Infrastructure). Ці технології забезпечують безпосередній обмін даними між транспортними засобами та між транспортними засобами та інфраструктурою відповідно, створюючи нові можливості для підвищення безпеки руху, ефективності транспорту та комфорту водіїв.

Основні особливості розумної мобільності в контексті V2V та V2I:

- Підвищення безпеки дорожнього руху:

- Раннє виявлення небезпек: Автомобілі можуть обмінюватися інформацією про аварійні ситуації, об'їзди, обмеження швидкості, що дозволяє водіям своєчасно реагувати та уникнути аварій.

- Покращення видимості: За допомогою радарів та камер транспортні засоби можуть "бачити" за перешкодами, що особливо корисно на поворотах та в умовах обмеженої видимості.

- Створення безпечних транспортних потоків: Системи V2V та V2I дозволяють координувати рух транспортних засобів, знижуючи ризик зіткнень та пробок.

- Збільшення ефективності транспорту:

- Оптимізація маршрутів: Автомобілі можуть отримувати інформацію про пробки, аварії та інші перешкоди в режимі реального часу, що дозволяє їм вибирати оптимальні маршрути.

- Зменшення споживання палива: Завдяки координації руху та оптимізації швидкості можна знизити споживання палива та викиди шкідливих речовин.

- Підвищення пропускної здатності доріг: Системи V2V та V2I дозволяють більш ефективно використовувати дорожню мережу, особливо в умовах високої щільності руху.

- Створення нових сервісів:

- Автономне водіння: Технології V2V та V2I є основою для розвитку автономних транспортних засобів.

- Інформаційні сервіси: Водії можуть отримувати різноманітну інформацію про дорожню ситуацію, паркування, заправки тощо.

- Нові бізнес-моделі: З'являються нові можливості для розвитку сервісів мобільності, таких як каршеринг, транспорт за викликом тощо.

- Інтеграція з іншими системами:

- "Розумні міста": Технології V2V та V2I можуть інтегруватися з іншими системами "розумних міст", такими як системи управління світлофорами, паркуванням, громадським транспортом.

- "Індустрія 4.0": Взаємодія транспортних засобів з іншими елементами виробничих процесів відкриває нові можливості для оптимізації логістики та управління виробництвом.

2.2.2. Особливості Індустрії 4.0

Індустрію 4.0 називають промисловою революцією сучасної епохи. Її цілі — отримати доступ до інформації про діагностику та продуктивність повних виробничих ліній з метою покращення їх ефективності або навіть якості продуктів/послуг, які вони виробляють [11].

Прогнозне обслуговування та промисловий Інтернет речей (IIoT) є ключовими компонентами еволюції промислової автоматизації до Industry 4.0. Мета полягає в модернізації існуючого промислового обладнання для досягнення автономної поведінки. Самоналаштування та діагностика дозволяють галузям скоротити витрати на технічне обслуговування та час простою машин, що призводить до загального покращення виробництва та безпеки.

Особливості цього домену також пов’язані з керуванням великою, але більш постійною/статичною кількістю пристроїв, які є частиною мережі. Зв’язок у режимі реального часу також є обов’язковою умовою промислових застосувань для покращення таких послуг, як віддалений моніторинг/ діагностика/обслуговування.

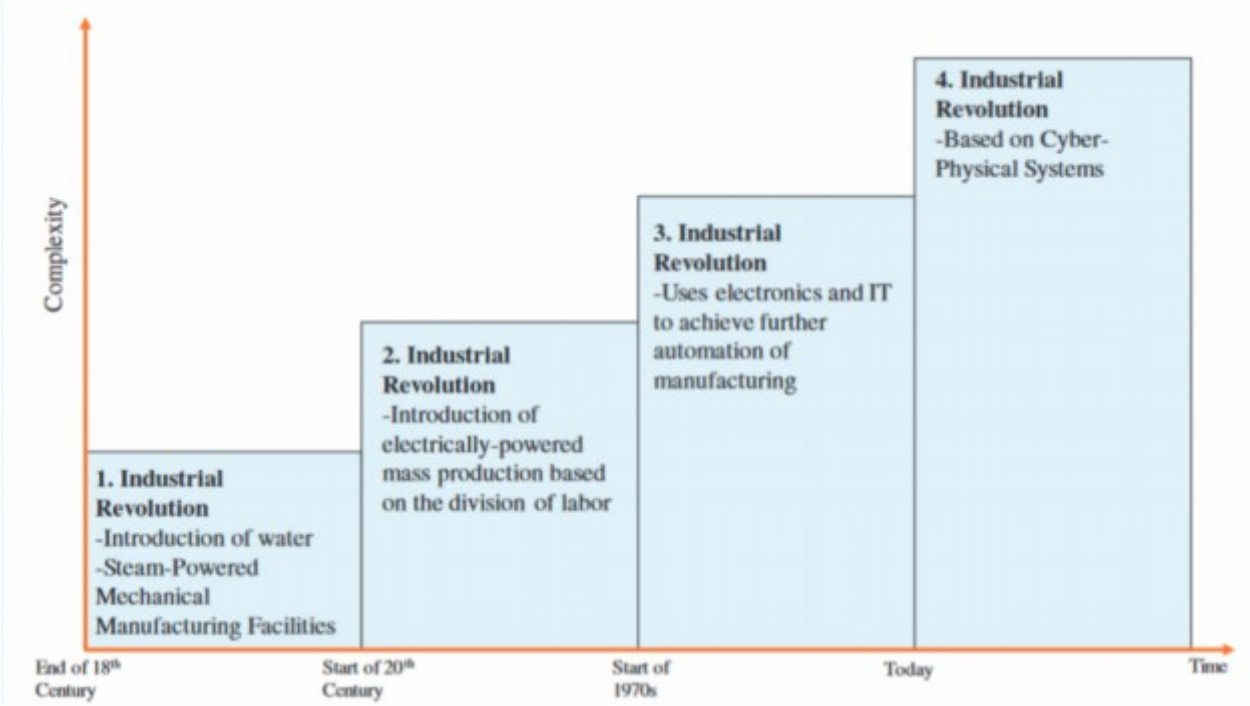

Рис. 2.4. Індустріальні революції

2.2.3. Пропонована восьмишарова архітектура

Нижче ми описуємо запропоновану архітектуру CPS 8C, як показано на рисунку 2.5.

Архітектура CPS 8C досягається шляхом додавання 3C-аспектів до архітектури CPS 5C. 3C-аспекти є коаліцією, клієнтом і контентом. Вони підкреслюють горизонтальну інтеграцію CPS, наприклад інтеграцію (або коаліцію) різних сторін та їх пов'язаної інформації (або контенту). Вони також підкреслюють найважливішу сторону, а саме клієнта, у процесі виробництва. Нижче ми описуємо три аспекти по одному:

- **Коаліція.** Цей аспект фокусується на інтеграції ланцюга вартості та інтеграції виробничого ланцюга між різними сторонами, залученими до

процесу виробництва. Різні сторони можуть спільно будувати ланцюг постачання та спільно планувати виробничі лінії для формування виробничого ланцюга з метою отримання певних продуктів гнучким та своєчасним способом. Якщо відбуваються будь-які коригування в процесі виробництва, різні сторони можуть спільно перебудовувати ланцюг постачання та виробничий ланцюг динамічно та своєчасно.

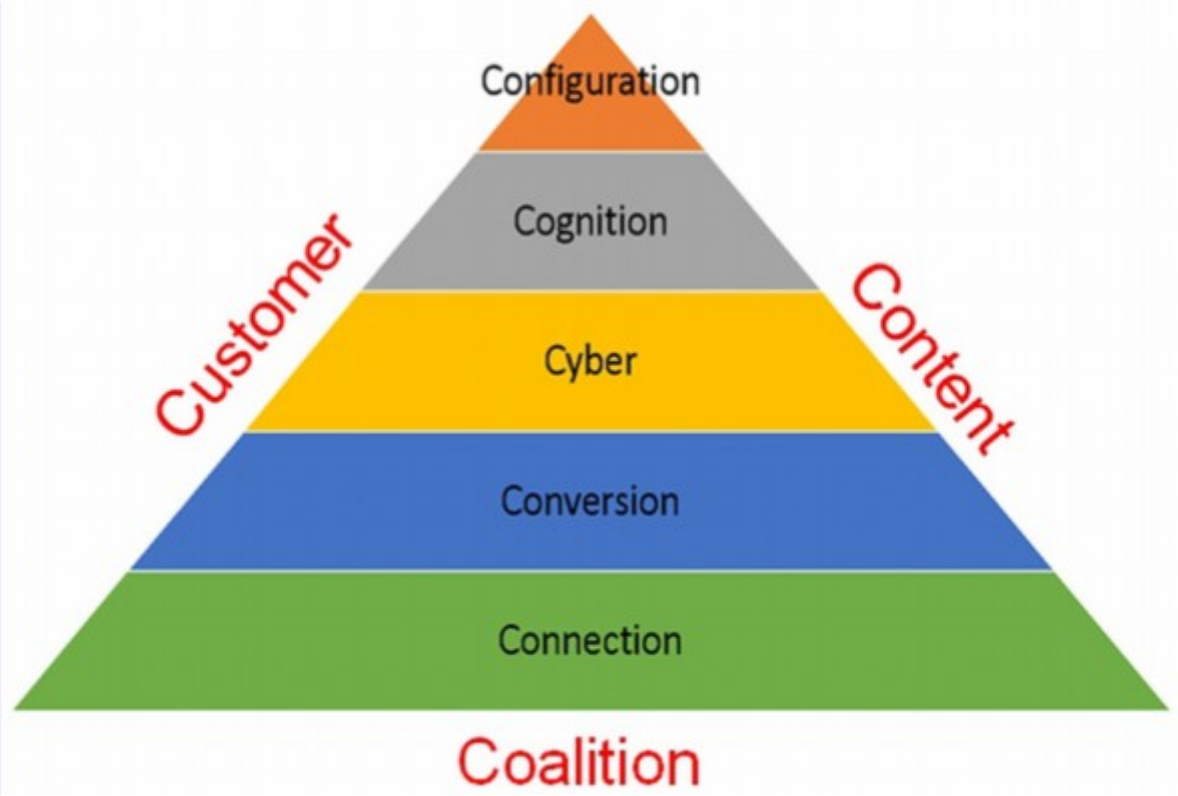

Рис. 2.5. Восьмишарова архітектура

- **Клієнт.** Цей аспект фокусується на ролі, яку клієнти відіграють у процесі проектування, виробництва та післяпродажного обслуговування продукту. Майбутні розумні фабрики можуть приймати різноманітні замовлення з невеликими обсягами від різних клієнтів та виконувати замовлення своєчасно. Клієнти, як агентство/оптовик або індивідуальний покупець, можуть брати участь у проектуванні продукту, відстежувати прогрес продукту та навіть змінювати специфікації продукту під час виробничого процесу. Це може бути досягнуто за допомогою концепції

виробництва, орієнтованого на продукт. Тобто фабрика може спонтанно реагувати на замовлення продукту, автоматично готуючи матеріали, гнучко плануючи виробничі процеси, динамічно переконфігуруючи виробничі лінії та автоматично організовуючи зберігання та доставку продукту. Таким чином, можна реалізувати як масову персоналізацію, так і масове виробництво. Клієнтів навіть можна повідомляти про прогрес виробництва, отримуючи електронні листи або текстові повідомлення. Розгляди в аспекті клієнта можуть відповідати на зміну від традиційної парадигми «масового виробництва» до нової, що виникає парадигми «масової персоналізації». Перша парадигма призначена для виробництва великої кількості продуктів однакової специфікації, тоді як друга призначена для виробництва різноманітності продуктів різної специфікації. Крім того, після доставки продукту клієнти можуть продовжувати отримувати післяпродажне обслуговування щодо експлуатації, використання, обслуговування та навіть переробки продукту для кращої якості досвіду.

- Контент. Цей аспект фокусується на видобуванні, зберіганні та запиті запису про трасування продукту. Вся інформація про виробництво, така як постачальники/джерела сировини, виробничі процеси, явища виробничого середовища (наприклад, температура, вологість, вібрація), виробничі параметри, склади та виробнича відправка, видобувається та зберігається належним чином як записи про трасування продукту для майбутніх запитів. Всі деталі післяпродажного обслуговування, такі як обслуговування, заміна деталей та інструкції щодо експлуатації, розглядаються як важливий контент та зберігаються своєчасно та належним чином у записах про трасування продукту. Більше того, всі деталі післяпродажного обслуговування, пов'язані з клієнтом, такі як обслуговування продукту, заміна деталей, усунення несправностей, переробка та скарги, пропозиції та коментарі користувачів у формі тексту, аудіо або навіть відео, розглядаються як важливі дані та зберігаються належним чином. Розгляди в аспекті контенту можуть допомогти досягти цілісного обслуговування продукту протягом усього його

життєвого циклу. Безумовно, аналізуючи всі збережені дані, можна покращити не тільки виробничий процес, конструкцію продукту та обслуговування клієнтів, але й передбачити тенденції ринку продукту.

2.3. Дослідження концепцій розумного будинку, розумної будівлі та розумної мережі

2.3.1. Розумний будинок

Концепція «розумного дому» спрямована на підвищення комфорту мешканців і водночас на економію енергії. Такі активи, як центральне опалення, посудомийна машина чи сигналізація, можна активувати дистанційно або навіть автономно керувати для безпечних, економічних цілей або продовження терміну служби.

Порівняно з іншими сферами Інтернету речей, розумний дім містить найменше, але також зростає кількість підключених пристроїв. Від розумних лампочок до інтелектуальних замків або розеток, різноманітні пристрої, виготовлені різними постачальниками апаратного забезпечення, повинні бути розміщені на одній загальній платформі. Це головний виклик для галузі IoT.

З точки зору застосування, у цьому полі IoT є більше можливостей, ніж, -наприклад, у розумних будівлях, якщо враховувати аспекти конфіденційності та безпеки.

Завдяки побудові внутрішньої комунікаційної мережі можна реалізувати мережу домашнього кондиціонування та інших розумних приладів за допомогою оптоволоконної мережі живлення. За допомогою інтелектуальних інтерактивних терміналів, розумних розеток, розумних приладів тощо ми досягаємо автоматичного збору інформації про електроенергію, аналізу, управління побутовими приладами; та побутові прилади досягають економічної експлуатації та контролю енергії [3, 4]. За допомогою телефону, мобільного телефону, Інтернету та інших засобів

система може дистанційно керувати домом та іншими послугами. За допомогою інтелектуальних інтерактивних терміналів ми також досягаємо виявлення диму, виявлення витоку газу, протипожежної охорони, аварійної допомоги та інших функцій домашньої безпеки, а також здійснюємо автоматичний збір та управління інформацією лічильників води, лічильників газу та підтримку та мережу головного вузла центру управління власністю, а також досягаємо авторизованої односторонньої передачі інформації про домашню безпеку та інші послуги. Рисунок 2.6 показує структуру розумного будинку.

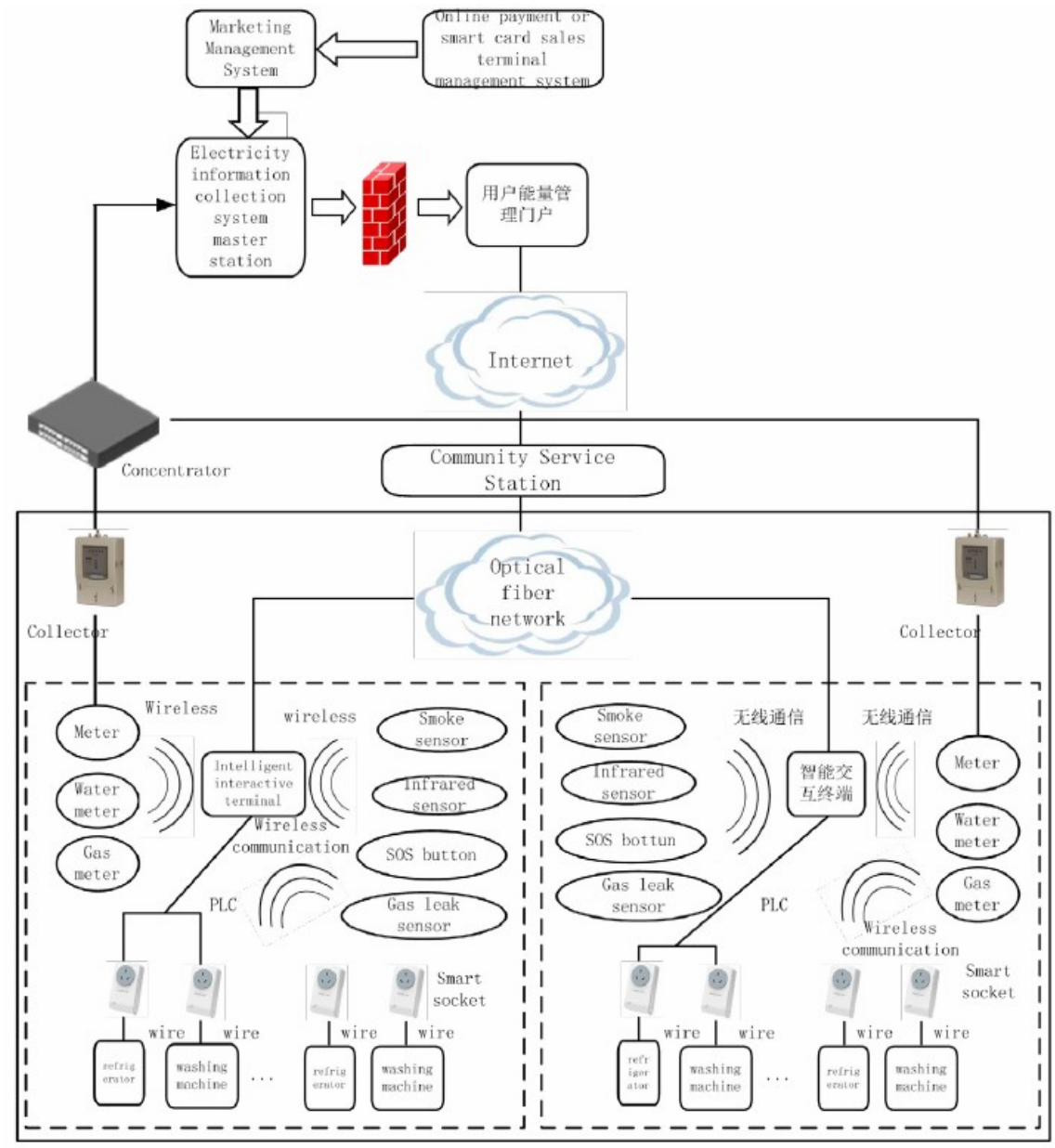

Рис. 2.6. Архітектура розумного будинку

2.3.2. Розумна будівля

Хоча концепція розумної будівлі подібна до розумного будинку, вона відрізняється більшою кількістю підключених пристроїв, а також проблемами конфіденційності та використання даних.

Концепція розумних будівель — це шлях до підвищення ефективності та зменшення витрат на споживання енергії в офісах, громадських і промислових будівлях. Ця мета досягається завдяки інтелектуальному управлінню приладами в будівлі. Дослідження, проведені в [1], показали, що системи ОВК є основним споживачем енергії в будівлях. У цій роботі також описано метод зменшення цього обсягу споживання енергії на 10-15% шляхом застосування комбінації магнітних і пасивних інфрачервоних датчиків .

Інтелектуальна система, представлена в [1], є ефективним першим кроком для досягнення важливої економії енергії. За допомогою датчиків стає можливим виявлення активності в приміщенні, а отже, можна відповідним чином налаштувати такі утиліти, як клімат-контроль та освітлення. Однак мета отримання енергозбереження не повинна перешкоджати комфорту людей, що користуються приміщеннями. Освітлення можна ввімкнути миттєво, не заважаючи кінцевому користувачеві, але вологість/теплова інерція накладає часовий інтервал модуляції до досягнення бажаних кліматичних умов. Крім того, хибні спрацьовування та негативні результати є значною проблемою, яка виникає під час впровадження цього рішення.

Локальності подій, датчиків і контролю недостатньо. Маючи додаткову інформацію щодо використання будівлі, бажаний стабільний стан можна підтримувати протягом цього інтервалу, але не більше. Наприклад, за допомогою доданої інформації з планувальника календаря клімат-контроль у кімнаті можна вмикати незадовго до початку зустрічі й до її завершення, замість того, щоб залишати її активною протягом усього дня або вмикати її точно тоді, коли особа (особи) увійти в кімнату. Іншим прикладом може бути

те, що агрегація за календарем також зменшить споживання енергії під час національних свят, якщо будівля вважатиметься закритою в ці дні.

Для моніторингу та керування потоком теплової енергії між тепловими підсистемами, використовується трирівнева архітектура сприйняття та прийняття рішень.

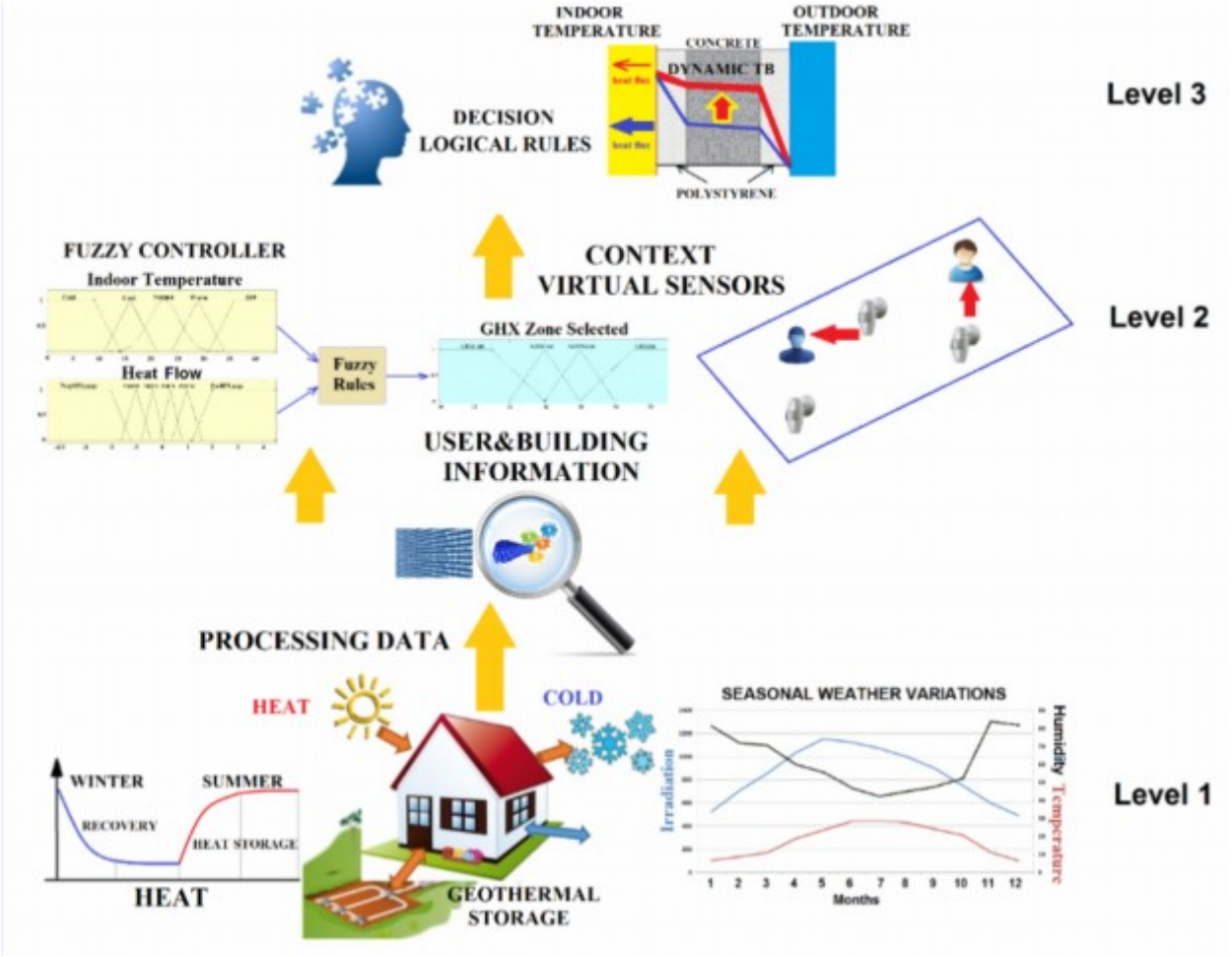

Рис. 2.7. Трирівнева архітектура прийняття рішень для управління потоком теплової енергії в розумній будівлі

Цикли сприйняття-керування спираються на дані, отримані від вузлів Інтернету речей (IoT). Вузли сприйняття IoT є датчиками присутності, температури та потоку, а вузли дії IoT пов'язані з електромагнітними клапанами та насосами. Різні інтерфейси використовуються для візуалізації, передачі інформації до баз даних та оновлення спільної пам'яті. Трирівнева архітектура сприйняття та прийняття рішень зображена на рисунку 2.7.

2.3.3. Розумна мережа

Класична енергомережа складається з функціональних активів, таких як електростанції, підстанції, трансформатори або розподільні лінії. На додаток до цієї топології, інтелектуальна мережа має мережу даних, яка ділиться відповідною інформацією для функціональних ¬наборів, а також ліній електропередач і трансформаторів, які забезпечують двонаправлений потік електроенергії. Такі пристрої, як інтелектуальні лічильники, інвертори, датчики напруги чи струму, дають уявлення про фактичний обмін електроенергією елементів у функціональній мережі. Крім того, такі дані, як інформація про прогноз погоди або ціноутворення в режимі реального часу на електроенергію, також актуальні для цієї теми, оскільки вони можуть впливати на фактичну поведінку біржі електроенергії.

Розумна мережа охоплює мережу розумних будівель разом із сторонами, що постачають енергію, і трейдерами. Метою інтелектуальної мережі також є підвищення ефективності та зменшення витрат (або збільшення прибутку), але в більшому масштабі.

Класична електрична мережа реагує на зміни навантаження в мережі. Іншими словами, електростанції регулюють свою потужність відповідно до миттєвого навантаження в мережі. Це неадекватний спосіб доставки енергії, оскільки ці модуляції вихідної потужності впливають на ефективність електростанцій або інших активів у мережі. Також є втрати потужності на транспортному шарі. Перше рішення, яке вже було впроваджено в більшості класичних електромереж, полягає в наявності двох тарифів на електроенергію як для споживання, так і для виробництва. Таким чином, люди заохочуються вирівнювати споживання енергії протягом дня, плануючи роботу таких приладів, як пральні чи посудомийні машини, на ніч.

У пропозиції інтелектуальної мережі навантаження електроенергії контролюється та прогнозується детально. У результаті електростанції можуть досягти вищої ефективності у виробництві електроенергії за рахунок збільшення часу наростання/зниження.

Це також означає, що для окремих учасників мережі можна запровадити краще управління енергією для зменшення витрат . Це робиться, наприклад, шляхом планування роботи приладів, щоб увімкнути, коли енергія дешевша.

Сучасне розгортання електромережі переходить від класичної до розумної. Що стосується клімат-контролю, деякі будівлі підключені до кількох централізованих теплових насосів.

Намір полягає в тому, щоб охопити якомога більше типів будівель для виробництва будівельних профілів. Типи будівель характеризуються засобами клімат-контролю (деякі мають газове опалення, інші покладаються на зовнішнє джерело опалення або електричне опалення), якістю ізоляції або корисною поверхнею. За допомогою будівельних профілів можна було б легше передбачити енергообмін для даної конструкції.

Ці будівлі ще мають бути підключені, але їх кінцева збірка реалізує мікророзумну мережу.

Основний варіант використання електричної мережі полягає в тому, щоб забезпечити краще розуміння споживання та виробництва енергії будівлею. Для підтвердження концепції було обрано кілька будівель, які будуть оснащені розумними лічильниками. Вони підібрані таким чином, щоб охоплювати широкий спектр характеристик будівель. Ці аспекти стосуються якості їх теплоізоляції, незалежно від того, чи є у них централізоване газове опалення чи ні, наявність відновлюваних джерел енергії (переважно фотоелектричних панелей). Буфери (батареї/ водневі баки) також можуть бути інтегровані в майбутньому.

Еволюція електроніки та телеметрії призвела до розробки різноманітних датчиків і приладів, здатних збирати інформацію з навколишнього середовища та надсилати її (по повітрю) на інші пристрої. Крім того, системи двосторонньої телеметрії відкрили можливість для цих пристроїв керувати дистанційно або навіть самі діяти як кластери для збору даних з інших пристроїв. Двосторонній зв'язок також розширюється до

використання приводів, які можуть контролювати або впливати на середовище на основі попередньо визначених алгоритмів.

Проте деталізація пристроїв є гнучкою. Під пристроєм можна розуміти простий датчик або виконавчий механізм, але також і більший вузол. Для теми розумної будівлі це може бути система управління будівлею (BMS). Ця система включає в себе формування та контроль над різними активами, такими як HVAC або системи освітлення.

2.4. Структура обміну даними в інтелектуальній мережі

У сфері цього проекту фреймворк визначається як платформа, яка дозволяє користувачам розробляти програми та взаємодіяти з доступними пристроями для досягнення цілей цих програм. Таким чином, структура обміну даними абстрагується від специфічних характеристик пристроїв і функціональності додатків виробника та зосереджується на налаштуванні даних, які передаються від одного об’єкта до іншого, і маніпуляціях, які з ними виконуються.

Рисунок 2.8 побудований на основі архітектур, наявних у літературі. Концепція віртуалізації пристроїв представлена в [12] і передбачає існування проміжного рівня програмного забезпечення, який полегшує взаємодію між додатками та пристроями високого рівня.

У [17] архітектура розділена на керування системою, керування пристроями та зберігання, усі з можливими локальними та хмарними реалізаціями, кожна зі своїми перевагами та недоліками залежно від доступного апаратного забезпечення та бажаної мети додатків.

Аналіз у [8] показує обмеження сумісності пристроїв і пропонує проміжне програмне забезпечення, відповідальне за інтеграцію різних протоколів зв’язку для вирішення цієї проблеми.

Щоб узагальнити різні концепції, доступні в літературі, і адаптувати їх до домену інтелектуальної мережі, ми представляємо загальну архітектуру

пристроїв розумної будівлі, структуру обміну даними та програми, показані на рисунку 2.8, організовані в різних рівнях абстракції. Цей рисунок допомагає підкреслити роль системи обміну даними.

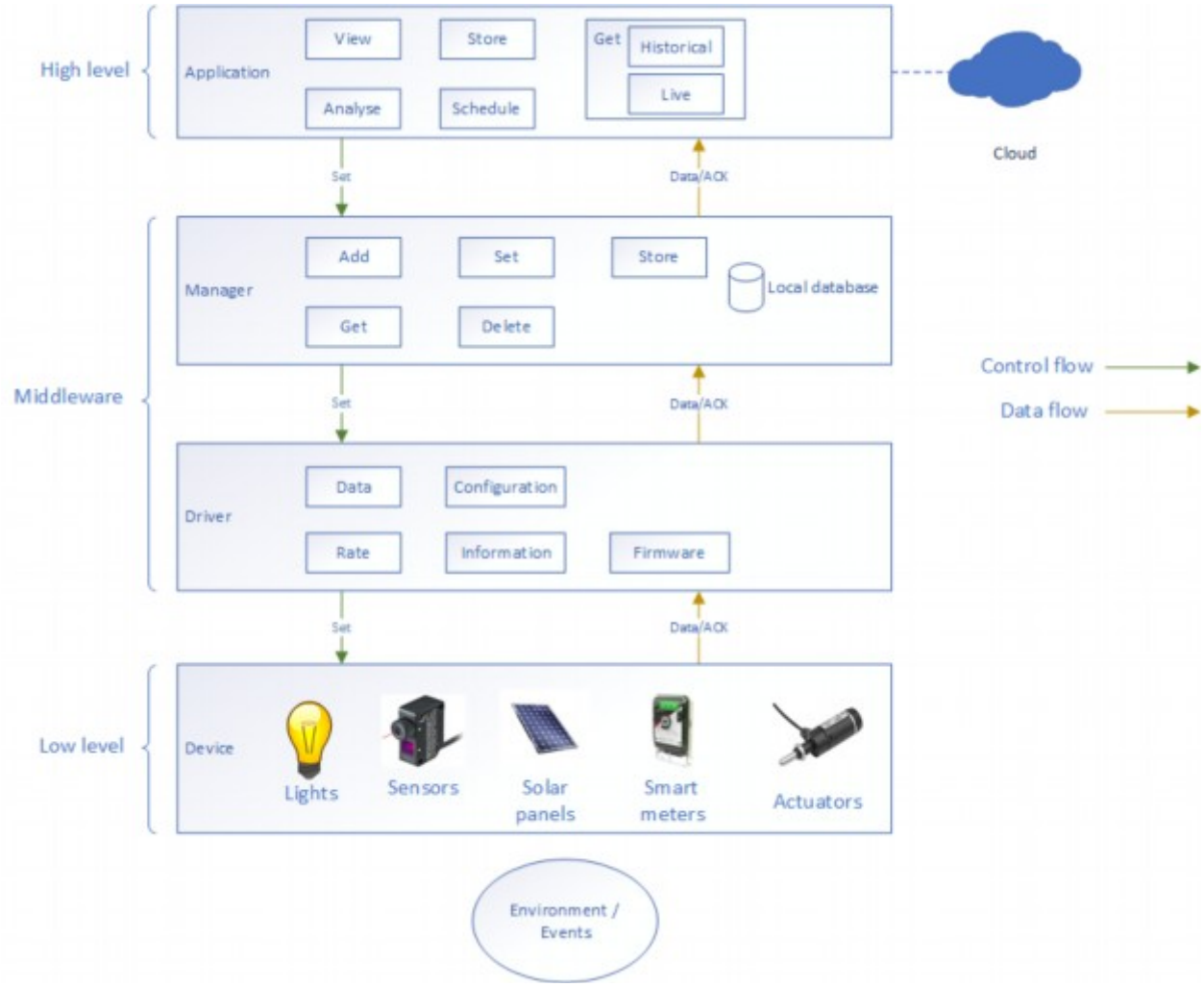

Рис. 2.8. Представлення приладів розумної будівлі, інфраструктури обміну даними та складання програм високого рівня

У нижній частині рисунку зображено зовнішнє середовище. Тут відбуваються різні події, які впливають на стан системи. Наприклад, температура може коливатися. Незалежно від того, чи відбувається це всередині будівлі чи зовні, це зміна стану, яка необхідна для застосування інтелектуального клімат-контролю.

Ці зміни стану навколишнього середовища реєструються пристроями, які є рівнем пристроїв архітектури. У попередньому прикладі це були б

датчики температури. Вони перетворюють зміну зовнішнього стану в машинно-інтерпретовані сигнали.

За абстракцією йде драйвер. Користувач головним чином відповідає за ідентифікацію та конфігурацію пристрою. Для датчика температури можна налаштувати частоту оновлення даних, що надсилається, і може бути доступна інформація щодо локалізації датчика, а також його характеристики. Менеджер може координувати роботу кількох користувачів. Розробка кластерів, додавання або видалення елементів з мережі, а також керування даними з пристроїв, зберігання та надання доступу до них є основними обов'язками цього рівня.

Програма виконує дії щодо наданих даних. Це можна зробити як віртуальною зміною (наприклад, викликом попередження/тривоги або реєстрацією події), так і фізичною (наприклад, увімкненням опалення чи відкриттям жалюзі). Розробник програми повинен знати про значення даних і доступність/ефект керованого перетворювача.

Слід зазначити, що сама структура зосереджена виключно на аналізі даних, передачі та взаємодії з пристроями та програмами. Основна увага в цьому проекті полягає не в тому, щоб реалізувати систему безпеки в рамках, наприклад, але остання повинна бути в змозі врахувати це, якщо це вимагається програмою. Однак це має бути визначено на рівні програми. Сучасний рівень техніки охоплює кілька концепцій, які можуть бути застосовані для реалізації програм.

2.5. Аналіз сучасних архітектур для реалізації

2.5.1. Монолітна архітектура

Монолітна архітектура означає збирання всіх необхідних компонентів програмного забезпечення в одному середовищі виконання. Таким чином, монолітна програма — це однорівнева програмна програма, де різні компоненти об'єднані в одну програму з однієї платформи [7].

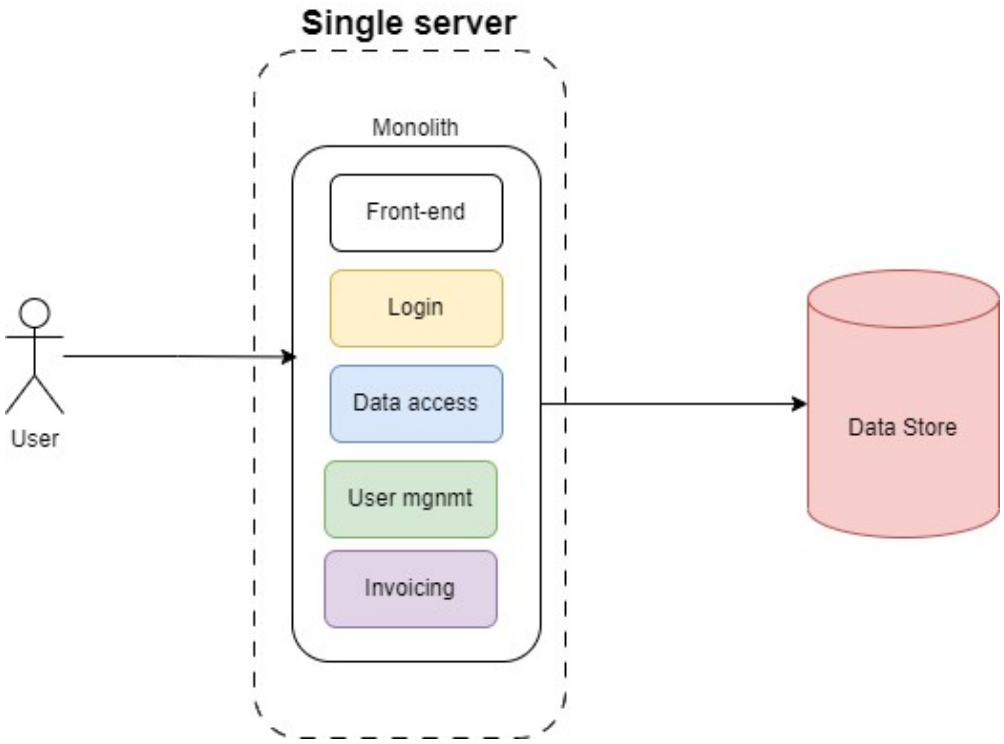

Рис. 2.9. Приклад монолітної архітектури

Прикладами цих компонентів можуть бути:

1. Обробники введення/виведення - відповідають за отримання вхідних даних від користувача та мають засоби відображення обробленої інформації.

2. Логіка програми – містить алгоритми та драйвери, які обробляють інформацію

3. Рівень бази даних – містить об’єкти доступу до даних, відповідальні за доступ до баз даних і керування ними

4. Інтеграція програми - відповідає за інтеграцію з іншими службами (наприклад, через обмін повідомленнями або REST API). Це також може бути інтеграція з будь-якими іншими джерелами даних.

Незважаючи на те, що програма складається з різних компонентів, вона створена та розгорнута як статична програма незалежно від платформи.

2.5.2. Сервіс-орієнтована архітектура

На відміну від залежності між компонентами, сервіс-орієнтована архітектура (SOA) — це архітектурна концепція, яка описує, як незалежні

служби можуть бути зібрані стандартизованим способом для надання функціональних можливостей, які задовольняють вимоги користувача програмного забезпечення. Сервіси, які також називаються компонентами архітектури, незалежні один від одного, навіть якщо зібрані разом, і можуть повторно використовуватися іншими учасниками мережі завдяки стандартизованому інтерфейсу [23].

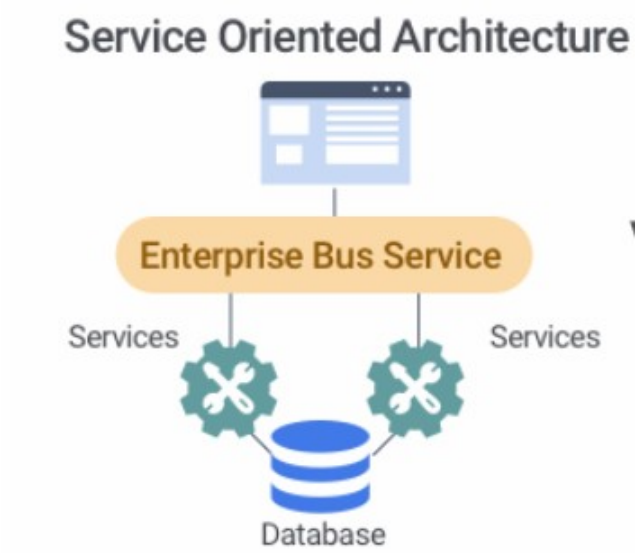

Рис. 2.10. Приклад сервіс-орієнтованої архітектури

Для системи SOA потрібна інфраструктура, яка забезпечує взаємодію та гнучкість у збиранні послуг. На практиці це досягається за допомогою мікросервісів [24], Enterprise Service Bus (ESB) [23] або Integration Platform as a Service (IPaaS), які зазвичай використовуються для хмарного середовища.

2.5.3. Мікросервісна архітектура

У [24] архітектура мікросервісу аналогічно визначається як архітектура REpresentational State Transfer (REST). Основною особливістю цієї загальної архітектури IoT є атомарність, на основі якої описується послуга, і взаємодія , яка -визначає можливості збирання цих елементів. По суті, архітектура мікросервісу розділяє загальні служби, які містить архітектура REST, на

кілька функціональних блоків, які можна повторно зібрати, щоб забезпечити однакові (або більше) функціональні можливості.

Як правило, ці служби мають зв’язок між собою без збереження стану, а корисне навантаження є простим форматом, який зазвичай виконується за допомогою повідомлень HTTP.

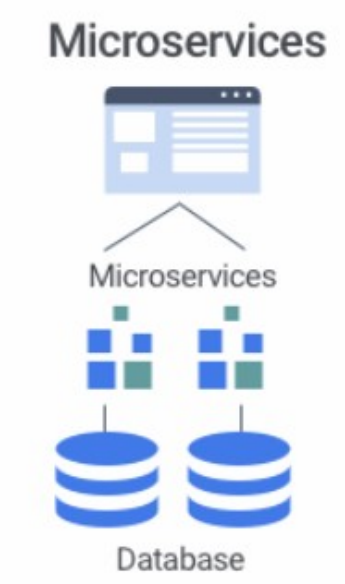

Рис. 2.11. Приклад мікросервісної архітектури

Логіка програми реалізується в кінцевих точках у цих службах, а структура використовується для транспортування даних. За визначенням, мікросервіс відповідає за збереження власних даних, а зв’язок з іншими мікросервісами здійснюється через HTTP та (здебільшого) REST без збереження стану. Через їх асинхронний зв’язок необхідний періодичний моніторинг стану працездатності мікросервісів.

Якщо мікросервіси — це концепція розподілених обчислень, Open Service Gateway Initiative — це архітектура додатків, що реалізує таку концепцію. Він вимагає віртуальної машини Java (JVM) для розміщення своїх служб і не призначений для інтеграції віддалених викликів процедур. Основною функціональністю цієї архітектури є її оперативне керування

модулями, які можуть дереєструвати, інсталювати, оновлювати або видаляти окремі компоненти програмного забезпечення [22].

Інтеграційна платформа як послуга (IPaaS) і корпоративна службова шина (ESB) є основними типами реалізацій SOA, які покладаються на складну інфраструктуру для реалізації послуг . У цих архітектурах кінцеві точки слабко пов’язані, і їх інтеграція вимагає мінімальних зусиль. Як варіант мікросервісів, логіка додатків у корпоративній службовій шині (ESB) знаходиться у структурі. Таким чином, вузли використовуються для простих операцій введення-виведення, а проміжне програмне забезпечення відповідає за розміщення всіх служб, які використовують ці операції.

Висновки до розділу

Розділ присвячений дослідженню методологічних основ застосування технологій Інтернету речей (IoT) в контексті інтелектуальних мереж. Було розглянуто концепції IoT, їхню роль у розумній мобільності та Індустрії 4.0, а також запропоновано восьмишарову архітектуру для підтримки розумних мереж. Окрему увагу приділено розумним будинкам, будівлям та мережам, які використовують IoT для підвищення енергоефективності та автоматизації. Досліджено структури обміну даними в інтелектуальних мережах та проведено аналіз сучасних архітектур для їх впровадження, що дозволяє забезпечити надійну інтеграцію технологій IoT.

РОЗДІЛ 3. ІМПЛЕМЕНТАЦІЯ ФРЕЙМВОРКУ ІНТЕРНЕТУ РЕЧЕЙ ДЛЯ ОБМІНУ ДАНИМИ В ІНТЕЛЕКТУАЛЬНИХ МЕРЕЖАХ

3.1. Розробка сценарію використання

Щоб відобразити додаткову цінність використання структури спільного використання даних, описано конкретний варіант використання. У центрі уваги цього випадку використання – розробники програмного забезпечення.

Варіант використання описує розробку додатка приладної панелі розумної мережі, як це описано в другому розділі. У такій програмі кілька джерел даних агрегуються, щоб надати кінцевому користувачеві огляд потоку електроенергії, доходів/витрат.

Щоб зрозуміти переваги впровадження фреймворку, описано конкретний варіант використання розумної будівлі. Інформаційна панель, що відображає споживання та виробництво електроенергії для будівлі, є прикладом життєздатного застосування.

Інтелектуальний лічильник, встановлений у будівлі, збирає інформацію про споживання різних активів, таких як освітлення, розетки або системи опалення, вентиляції та кондиціонування, а також про виробництво фотоелектричних панелей. Локальний вузол IoT, наприклад Raspberry Pi, можна підключити до інтелектуального лічильника, а також до Інтернету для передачі доступної інформації. Дані зберігаються в базі даних і також використовуються як вхідні дані для живого графіка для кінцевого користувача.

Діаграма послідовності такої інформаційної панелі показана на рисунку 3.1 . Можна помітити, що існує кілька фізичних компонентів, які взаємодіють один з одним. Крім того, є деякі дії, які можна виконувати паралельно, пропонуючи незалежні функції, які можна використовувати одночасно.

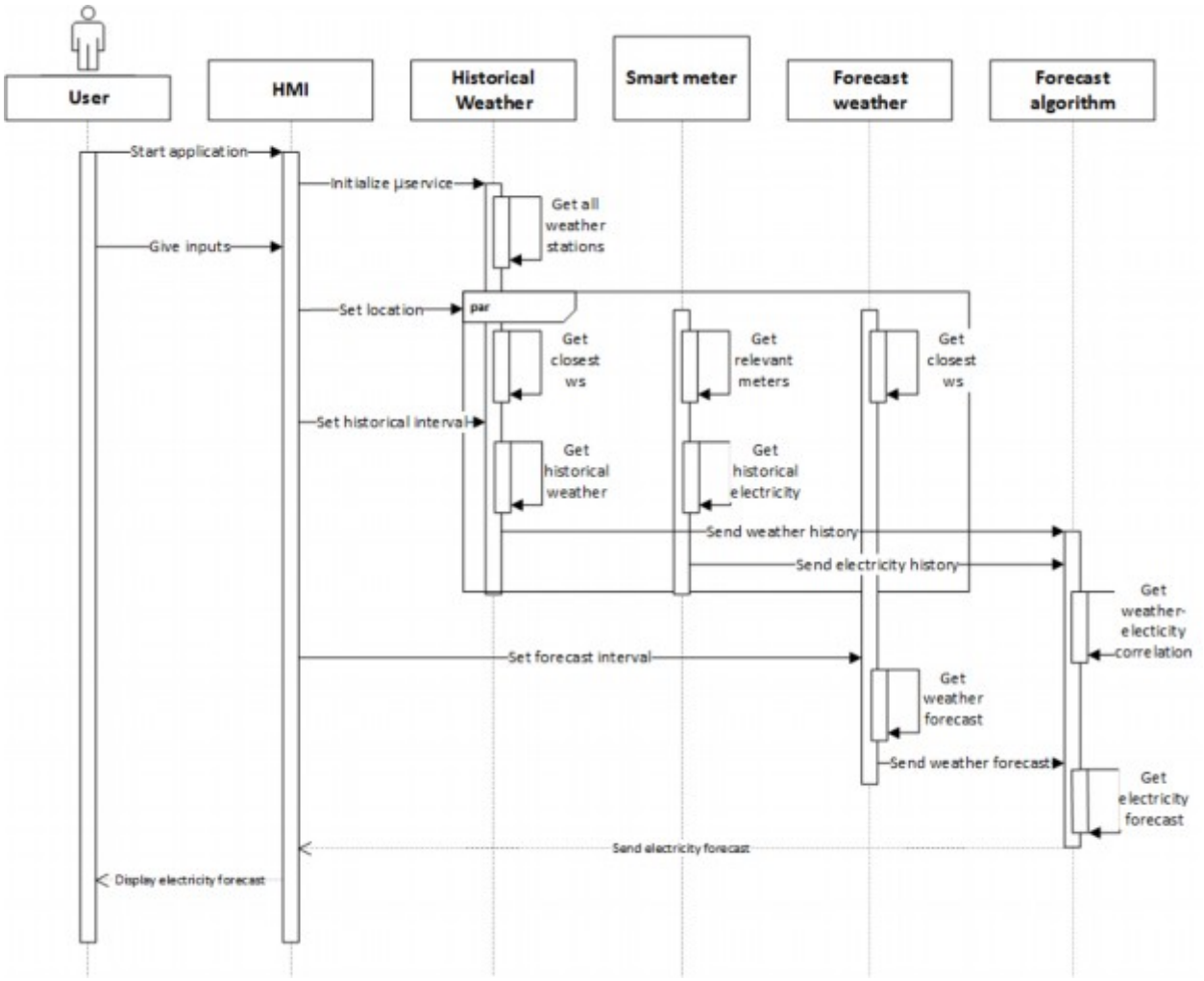

Рис. 3.1. Діаграма послідовності для прикладного додатка панелі керування

Наведена діаграма описує процес прогнозування споживання електроенергії. Вона показує, як різні компоненти системи взаємодіють між собою для отримання прогнозу.

Основні компоненти системи:

- Користувач: Ініціатор процесу, який вводить необхідні дані.

- HMI (Human-Machine Interface): Інтерфейс, через який користувач взаємодіє з системою.

- Історична погода: База даних з історичними метеорологічними даними.

- Розумний лічильник: Пристрій для збору даних про споживання електроенергії.

- Прогноз погоди: Сервіс, який надає прогнози погоди.

- Алгоритм прогнозування: Модель, яка використовується для прогнозування споживання електроенергії на основі історичних даних та прогнозу погоди.

Послідовність дій наступна:

- Ініціалізація: Користувач запускає програму, і система готується до роботи.

- Введення даних: Користувач вводить необхідні дані, такі як місцезнаходження.

- Збір даних про погоду: Система запитує всі доступні метеостанції та вибирає найближчу до вказаного місцезнаходження.

- Збір історичних даних: Збираються історичні дані про погоду та споживання електроенергії за певний період.

- Аналіз даних: Алгоритм аналізує зібрані дані, щоб виявити залежність між погодою та споживанням електроенергії.

- Прогноз погоди: Система отримує прогноз погоди на певний період.

- Прогноз споживання: На основі аналізу історичних даних та прогнозу погоди алгоритм розраховує прогноз споживання електроенергії.

- Виведення результату: Отриманий прогноз відображається користувачу.

Система використовує історичні дані про погоду та споживання електроенергії для створення моделі, яка дозволяє прогнозувати майбутнє споживання на основі прогнозу погоди. Такий прогноз може бути корисним для планування енергоспоживання та оптимізації витрат.

Можливі застосування:

- Домашні господарства: Для оптимізації споживання електроенергії та зниження витрат.

- Комерційні підприємства: Для планування енергоспоживання та управління витратами на електроенергію.

- Енергетичні компанії: Для прогнозування навантаження на мережу та оптимізації виробництва електроенергії.

Цей варіант використання використовується для оцінки переваг фреймворку. Можуть існувати інші випадки використання, коли додаток не можна розділити на окремі служби, і де структура може стати непотрібними накладними витратами. Фреймворк повинен надавати розробникам додатків свободу також реалізовувати рішення як є з невеликими накладними витратами.

3.2. Принципи проектування системи

Щоб отримати бажану поведінку системи, необхідно зробити огляд варіанту використання. Цей огляд охоплює те, як розробники виконують ітерації, щоб отримати кінцевий результат, а також як кінцеві користувачі взаємодіють із ним і який життєвий цикл програми.

Кілька принципів проектування виведено в [14]. Відповідні з них узагальнено для поточного випадку використання. Цих принципів слід дотримуватися не лише для поточної програми інформаційної панелі, але й для всіх рішень, пов’язаних із сферою Інтернету речей.

1. Можливість повторного використання додатка – слід сприяти тому, щоб додаток складався з простих будівельних блоків, щоб його можна було легко (повторно) зібрати. Це також означає, що розробник програми абстрагується від низькорівневої реалізації ¬різноманітних функціональних можливостей (наприклад, запитів HTTP/CAN) і вимагає лише заповнення джерел/приймачів даних і алгоритмів обробки.

2. Гнучкість програми – програма має різні версії розгортання залежно від уподобань зацікавлених сторін.

Наприклад, користувач, який безпосередньо взаємодіє з інформаційною панеллю, хотів би мати швидку та чуйну програму. З іншого боку, якщо той самий додаток розгортається в мережі з обмеженою пропускною здатністю, буде краще, щоб додаток був максимально ефективним з точки зору передачі даних, з обмеженими накладними витратами на зв’язок.

3. Масштабованість мережі – програма адаптується до масштабованої кількості джерел/приймачів даних і процесорів, доступних у мережі. Працездатність програми повинна бути гарантована в будь-який час.

4. Гнучкість і масштабованість мережі . Додаток адаптується до різних вузлів мережі, які приєднуються або залишають мережу без попереднього повідомлення.

а. Програма адаптується відповідно до вимог розробника програми щодо різних QoS (наприклад, затримка, споживання енергії)

б. Надійність програми має бути гарантована. Іншими словами, система повинна задовольняти вимоги програми з огляду на топологію мережі, реалізовувати резервні сценарії або відображати неможливість зробити це.

5. Розділення інтересів - програма повинна бути розроблена незалежно.

а. Розробники обладнання повинні надавати пристрої, з якими можна взаємодіяти через різні протоколи зв’язку.

б. Розробники додатків повинні бути стурбовані забезпеченням надійних алгоритмів, встановленням системних вимог і залежностей.

3.3. Функціональні та нефункціональні вимоги до структури обміну даних

У цьому розділі на основі аналізу представлені функціональні та нефункціональні вимоги, яким має задовольняти структура обміну даними. У підрозділі 3.3.1 описано поведінку фреймворку разом із його можливостями і обмеженнями, а у підрозділі 3.3.2 описуються найважливіші нефункціональні вимоги фреймворку.

3.3.1. Функціональні вимоги до поведінки фреймворку

Функціональні вимоги охоплюють бажану поведінку системи та виводяться з аналізу вимог [19]. Аналіз зацікавлених сторін і проблем схвалює визначення цих функціональних вимог у нашому випадку.

Досліджуваний фреймворк повинен мати можливість взаємодіяти як із прикладним, так і з рівнями пристроїв із мінімальними можливими витратами та без перешкод для функціональності жодного з них.

Сама структура не повинна перешкоджати розробці або розгортанню програм. Фреймворк повинен надавати інструменти та послуги, які можна використовувати для створення рішень. Їх мета полягає в тому, щоб відокремити розробників додатків від низькорівневих механізмів, які беруть участь у досягненні цієї мети і не мають відношення до самої логіки додатків.

Наступні вимоги до структури випливають з аналізу проблеми, а також опису сценарію використання, наведеного в розділі 3.1.

R1. Фреймворк повинен мати можливість запускати (розподілені) програми на пристроях.

R2. Функціональні вимоги та атрибути якості, встановлені додатками, також повинні бути відображені в фреймворку.

R3. Структура повинна пропонувати базові функціональні можливості для розробки та розгортання додатків, такі як:

а) Конструктивні перевірки

б) Настанови

R4. Структура повинна пропонувати базові функції для обміну даними, такі як:

а) Управління доступом / Безпека / Контроль даних

б) Логіка даних / Зміна потоку даних

R5. Структура повинна мати мінімальні накладні витрати на конфігурацію та час виконання програм і пристроїв.

Ми визначаємо наведені нижче передумови, щоб додатки та пристрої відповідали цим вимогам.

Для R1, щоб платформа могла розміщувати пристрої та запускати програми, визначено такі вимоги:

S1. Пристрої в мережі повинні бути ідентифіковані за унікальним ідентифікатором і повинні мати інтерфейс зв’язку з іншими вузлами.

S2. Програма повинна бути розділена на програмні компоненти, які можна повторно збирати та змінювати окремо.

S3. Програмні компоненти програми можуть бути налаштовані на поточний запуск, якщо взаємозалежність для їх виконання не вказана явно.

S4. Якщо в мережі, за запитом вузол повинен повідомити про свій статус (доступний/виконується).

Нефункціональні вимоги для R2 визначено наступні:

S5. Програма повинна мати атрибути якості, які піддаються кількісному вимірюванню (наприклад, пропускна здатність, затримка, використання пропускної здатності).

S6. Додаток має мати функцію збору / маніпулювання даними або керування пристроєм.

S7. Додаток має оновити свої вимоги щодо необхідних ресурсів після зміни.

Для задоволення R3a та R3б щодо полегшення розробки та перевірки застосунків визначено такі попередні умови:

S8. Пристрої в мережі повинні надавати інформацію щодо своїх апаратних специфікацій, якщо це доступно/застосовується. (наприклад, датчики/виконавчі механізми, обчислювальна/ємність зберігання, оперативна пам'ять, сила сигналу мережі, доступність вузлів)

S9. Пристрої повинні мати можливість переривати виконання застосунку для цілей синхронізації мережі.

S10. Пристрої повинні дозволяти (сертифіковану) зовнішню конфігурацію з точки зору підключення та виконання.

Для задоволення R4a та R4b, що стосуються контролю та управління даними, визначено такі попередні умови:

S11. Розробник застосунку може налаштовувати зміст та доступність даних на пристроях залежно від потреб інших зацікавлених сторін.

S12. Пристрої повинні мати можливість змінювати свій час виконання, якщо сертифікати змінилися. Тобто пристрій повинен заборонити надсилання інформації іншому пристрою, якщо йому наказано припинити цю передачу даних.

Для задоволення вимоги R5 щодо опису системи, яка вимагає, щоб платформа мала мінімальний вплив на продуктивність застосунків, встановлено таку попередню умову:

S13. Застосунки та пристрої повинні мати можливість конфігуруватися.

3.3.2. Нефункціональні вимоги

Платформа повинна мати метрики якості обслуговування (QoS), а також засоби покращення послуг. Їх можна кількісно оцінити за допомогою вимірювань, визначених у цьому підрозділі.

Існує багато властивостей, які можуть оцінити відповідність платформи для різних областей IoT. Однак наступні є актуальними для концепції розумного будинку. Також представлено визначення та кількісне визначення цих атрибутів якості. Ортогональність цих атрибутів також є окремою темою аналізу.

1. Налаштовуваність

Налаштування в цьому контексті означає здатність платформи аналізувати або збагачувати дані відповідно до потреб розробника.

Повинно бути можливим інтегрувати будь-які служби, які відповідають як з точки зору застосунку, так і з точки зору пристрою. Іншими словами, платформа не повинна перешкоджати повному розв'язку. Вузькі місця, якщо вони присутні, повинні бути максимум на рівні застосунку або пристрою та відображатися відповідно.

Наприклад, якщо пристрій має лише представлення з плаваючою точкою для своїх даних, платформа повинна надавати цю інформацію на рівень застосунку також як представлення цілого числа, якщо це запитуваний формат.

Кількість можливостей аналізу на функцію є одиницею виміру для цього атрибуту.

Налаштування може вплинути на продуктивність застосунків, оскільки певні перетворення можуть вимагати апаратного віртуалізації. Для перетворення з плаваючою точкою, якщо пристрій не має апаратних модулів, спеціально розроблених для цієї мети, таке перетворення може вимагати занадто багато циклів виконання для мети застосунку.

2. Модульність

Модульність – властивість системи розділятися на незалежні функціональні блоки, які мають інтерфейси.

Фреймворк повинен мати прості методи та інтерфейси, щоб з ним можна було працювати одночасно. Ця характеристика також дозволяє розгортати структуру на кількох пристроях, щоб усунути SPF, увімкнути масштабування та розподілені обчислення. Цей атрибут якості (QA) можна кількісно визначити кількістю модулів, з яких може складатися дане рішення, їхньою здатністю функціонувати незалежно та видом розробки для ілюстрації попередніх двох.

Модульність відбувається за рахунок складності систем і непотрібних витрат на продуктивність для невеликих програм.

3. Композиційність

Компоновність відноситься до властивості фреймворка збирати різні сценарії з використанням однакових компонентів.

Фреймворк має використовувати доступні елементи в системі. Він повинен відображати доступність пристроїв у мережі в режимі реального часу з обох сторін рішення та перерозподіляти його завдання, якщо це необхідно. Кількісну оцінку цього атрибута можна виконати за кількістю сценаріїв композиції, які може підтримувати фреймворк.

Компонування відбувається за рахунок стандартизації, оскільки підвищується складність системи та накладні витрати через додаткові

перевірки, які необхідно виконувати на програмних і апаратних компонентах.

4. Конфігурація

У цьому контексті конфігурованість стосується атрибута фреймворку для зміни функціональності його компонентів під час виконання.

Фреймворк повинен надавати програмі можливість налаштовувати її компоненти під час виконання. Наприклад, швидкість передачі даних датчика можна перемикати під час виконання таким чином, щоб він використовував меншу пропускну здатність мережі за запитом. Цей QA можна оцінити за здатністю інфраструктури адаптувати конфігурацію своїх компонентів під час виконання.

Конфігурованість залежить від ефективності, оскільки пристрої повинні мати засоби доступу до середовища зв'язку, виділяти для цього час і переривати/змінювати час роботи, якщо потрібно.

5. Надійність

Надійність означає здатність інфраструктури підтримувати послуги, які вона надає, у разі випадкових збоїв вузлів. Це також можна вважати резервним QA. Структура повинна мати резервні засоби для послуг, які вона надає. Наприклад, якщо вузол кластера виходить з ладу в мережі, його топологію слід визначити повторно, щоб забезпечити достатній функціональний потік. Це можна кількісно визначити кількістю сценаріїв збою, які він може обробити, або поведінкою системи, коли різні вузли раптово виходять з мережі. Надійність шкодить затримці та пропускній здатності.

6. Безпека

Безпека — це властивість структури, яка дозволяє лише авторизованим користувачам отримувати доступ до даних/змінювати їх. Фреймворк повинен мати засоби встановлення індивідуальних прав взаємодії відповідно до рівнів привілеїв користувача. Структура також повинна мати можливість реєструвати та перевіряти порушення безпеки, а також виявляти неетичне

використання наданих даних/активів. Структура повинна підтримувати реалізацію (загальних) стратегій безпеки, таких як шифрування або сертифікати, але не примусово їх використання. Цей атрибут можна кількісно оцінити, перевіривши доступну інформацію, що а користувач має відповідно до своїх облікових даних і можливості інших (неавторизованих) сторін отримати доступ до цих даних або змінити їх. Контроль якості безпеки впливає на пропускну здатність і затримку програм, оскільки автентифікація та дешифрування даних потребують додаткових циклів виконання.

3.4. Представлення концепції застосування Інтернету речей для обміну даними в інтелектуальних мережах

Масштабованість є ключовим фактором у високопродуктивних додатках IoT, і, враховуючи прогнозоване вибухове зростання цього домену, важливо розробити структуру, яка дає змогу здійснювати моніторинг, захист і керування великою кількістю пристроїв, які використовують спільні ресурси. З точки зору програмного забезпечення, масштабованість може бути досягнута шляхом впровадження архітектури без збереження стану, де немає необхідності підтримувати інформацію кожного пристрою від одного запиту до іншого. Крім того, архітектура мікросервісів уможливить масштабування, оскільки кілька функціональних одиниць можуть працювати незалежно один від одного [15].

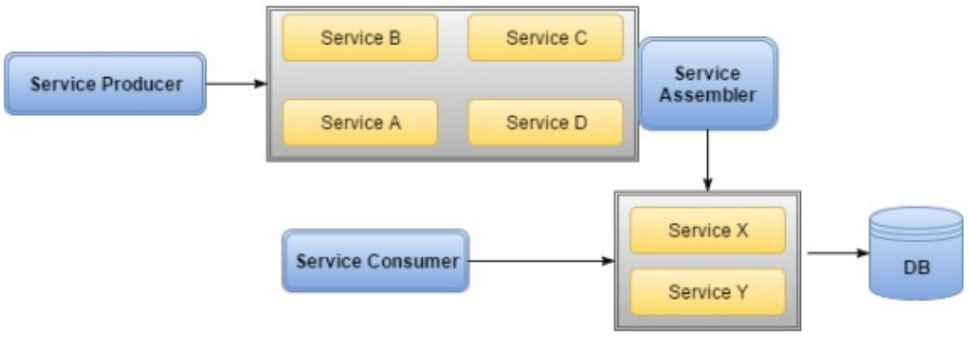

Рис. 3.2. Представлення мікросервісу для IoT Service Atomity

Враховуючи велику кількість пристроїв, які є частиною мережі IoT, основні проблеми для забезпечення надійної платформи полягають у можливості зберігання та доступу до даних, одночасно забезпечуючи узгодженість інших функціональних можливостей системи. Маючи мікросервісну архітектуру, кожен компонент використовуватиметься лише для однієї конкретної потреби, гарантуючи, що кожен раз, коли компонент викликається програмою, він забезпечуватиме надійну та узгоджену службу.

На основі аналізу, проведеного в попередніх пунктах, можна визначити вдосконалену платформу. Вихідним пунктом для розробки платформи є робота, виконана в [18].

Перевагами цієї платформи є розділення функціональності та управління на верхньому рівні. Недоліками цієї платформи є її статичне розгортання та відсутність налаштування. Ці два недоліки можна пом'якшити шляхом реалізації динамічного механізму розподілених обчислень. Ця концепція вже представлена в [24] під терміном мікросервісів.

Функції управління, зберігання та контролю також повинні бути включені в платформу. Аналіз поточних платформ IoT виявив необхідність реалізації легкої платформи, яка є модульною, компонувальною та надійною.

Також очевидно, що швидка розробка та розгортання, а також повторне використання є ключовими атрибутами якості для розробленої платформи через швидке розширення підключених пристроїв у всіх областях IoT.

Мікросервіс моніторингу. Запропоноване рішення має актор моніторингу, який може визнавати можливі, запитувані та доступні послуги. Це також стандартна характеристика архітектури мікросервісів. Крім того, для цілей цього проекту цей мікросервіс також відповідає за (пере)розгортання застосунків шляхом оновлення своєї мережі за певних умов. Він діє як арбітр між пристроями та застосунками. Завдяки інтеграції стандартизованих програмних компонентів також можна реалізувати RPC, долаючи основне обмеження мікросервісів.

Більше того, залежно від мети зацікавленої сторони, функціональність пристроїв у мережі може бути налаштована. Ті самі функціональні можливості можна досягти за допомогою кількох конфігурацій, кожна з яких має свої переваги та недоліки.

Оркестратор. Якщо актор моніторингу забезпечує функціональність застосунку, то оркестратор забезпечує досягнення встановлених користувачем вимог до якості (QA).

Рисунок 3.2 зображує таку систему. Можна побачити, що кілька служб можуть бути зібрані різними способами для обслуговування необхідної мети. Великою перевагою цієї системи є те, що вона забезпечує гнучку та масштабовану асемблі. Існує 3 сценарії, які можуть бути викликані, де додавання цього оркестратора служб буде корисним для платформи.

У випадку застосунку, який вимагає короткого часу відповіді, асемблер може приписати всю службу одному актору (який має достатню обчислювальну потужність). Таким чином, уникнути накладних витрат на комунікацію.

У разі протилежної ситуації, коли служба вимагає високої обчислювальної потужності на вузлі з низькою продуктивністю та накладні витрати на комунікацію є незначними з точки зору пропускної здатності та споживання енергії, обчислення можна делегувати.

Другий сценарій - це сценарій складного завдання, яке вимагає великої кількості (малих) обчислень, таких як усереднення або підсумовування. Завдяки розподіленню обчислень для кількох вузлів у мережі, середній/гірший час відповіді всієї системи повинен бути покращений. Це залежить, звичайно, від накладних витрат на комунікацію між вузлами також.

Третій і останній сценарій, де ця установка була б корисною, є випадок, коли є SPF для заданого застосунку. У цьому випадку платформа зможе застосувати надлишковість до системи та забезпечити доступність служби.

Відсутність налаштування під час виконання може бути додатково використана з використанням спеціалізованої функціональності для взаємодії з пристроями. Ця функціональність вимагає, щоб сам пристрій мав цю можливість бути налаштованим. Наприклад, у випадку застосунку, який вимагає великого обсягу даних протягом певного часу, застосунок повинен мати можливість налаштувати пристрій відповідно. В результаті навантаження на мережу буде зменшено. Інша можливість зменшити передачу даних у мережі полягає в тому, щоб платформа мала дизайн для зміни опитування даних. Це передбачає необхідність платформи змінити конфігурацію нижнього рівня під час виконання.

Завдяки додаванню оркестратора мікросервісів та налаштовуваності під час виконання, а також метрик якості обслуговування та системи відновлення, ця платформа буде підходити для теми розумного будинку/мережі.

На високому рівні абстракції послугу можна розглядати як збірку компонентів, як показано на рисунку 3.3.

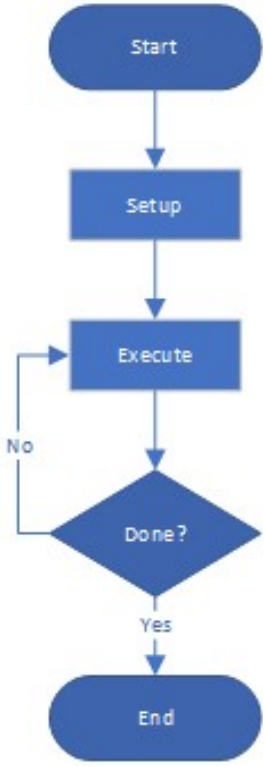

Рис. 3.3. Загальна блок-схема для послуг

Служба ініціалізується, виконує своє завдання після завершення, а потім завершує роботу. Може статися так, що служба повторно ініціалізується під час виконання. Перевагою цього підходу є фреймворк на основі незалежних компонентів, де сервіси мають гнучкість роботи незалежно.

Є кілька компонентів, які повинні працювати в заданій послідовності, щоб отримати бажаний результат програми. Деякі компоненти є специфічними для даних рішень, інші є загальними. Тим не менш, всі вони можуть бути використані повторно. Щоб зрозуміти різницю між загальними та конкретними компонентами та те, як фреймворк бере участь у цьому розрізненні, аналізується конкретний варіант використання.

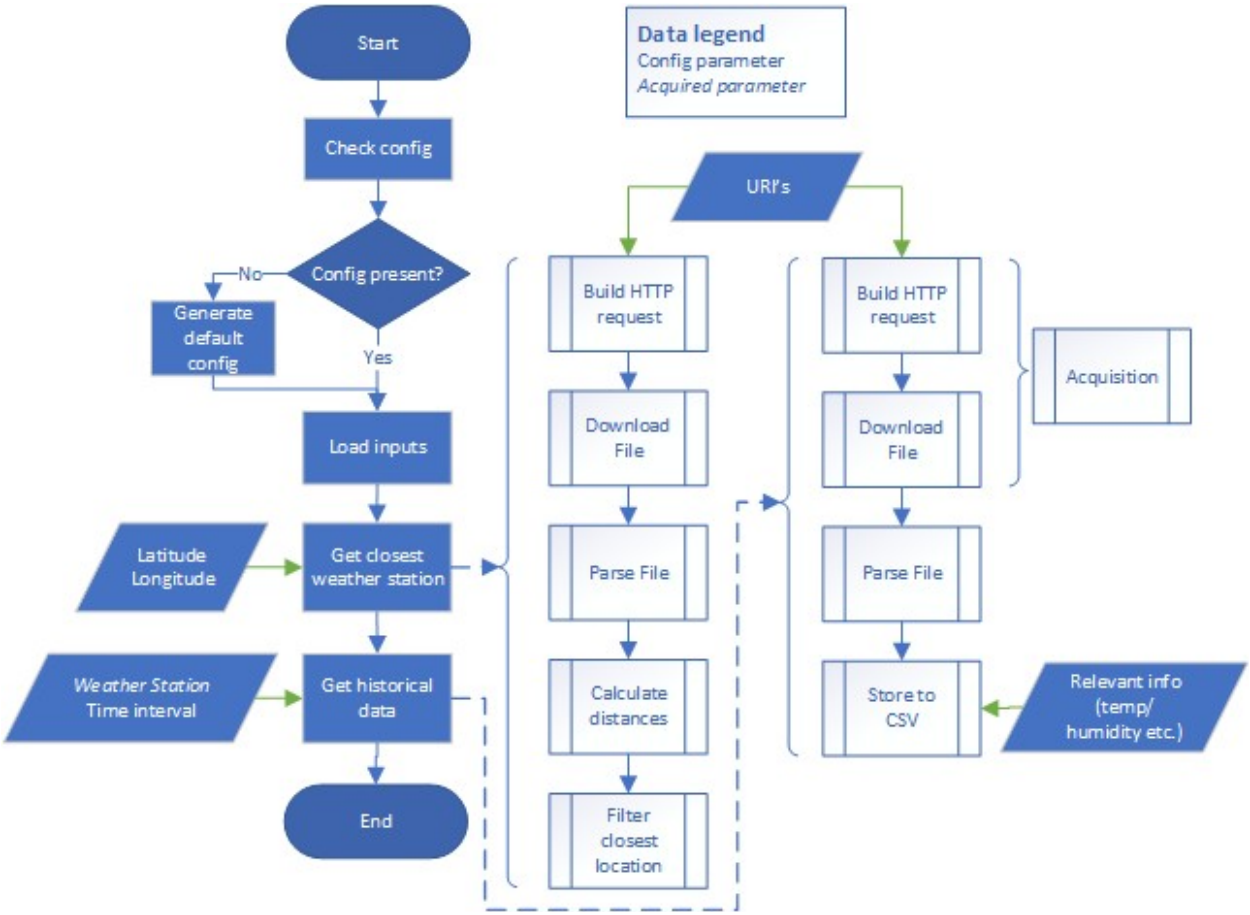

Рис. 3.4. Сервіс для зберігання історичних даних про погоду

Наступний варіант використання описує програмні компоненти, які виконуються послідовно для зберігання історичних даних певної

метеостанції. Зліва на рисунку 3.4 показано блок-схему цього рішення. Для кожного виконання рішення існує фаза налаштування, на якій аналізуються певні вхідні дані.

Є два специфічні компоненти, які характерні для цієї послуги, а саме отримати найближчу метеостанцію та отримати історичні дані.

У правій частині блок-схеми показано наступні компоненти, які містить ця служба. Ці компоненти можна повторно зібрати в іншому порядку, або деякі з них можна навіть пропустити для інших служб.

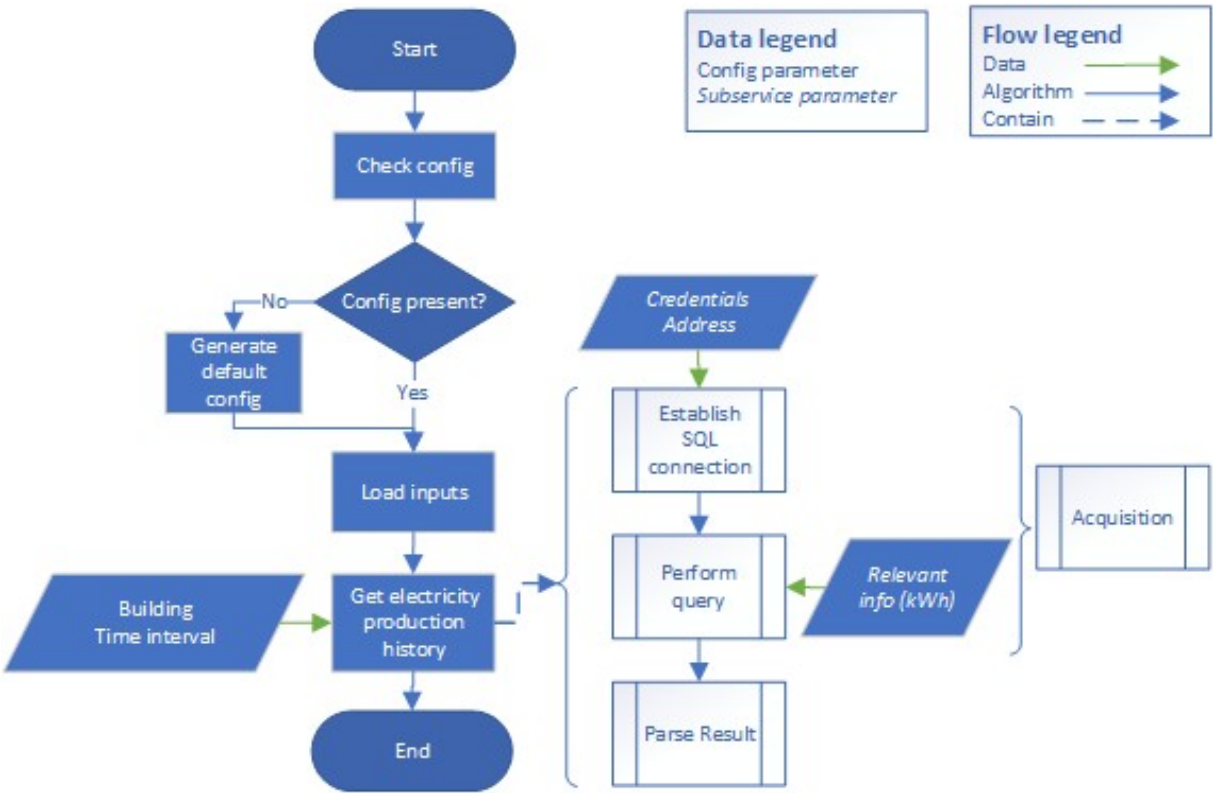

Рис. 3.5. Алгоритм сервісу для отримання даних про споживання електроенергії за попередні періоди

На рисунку 3.5 показано блок-схему іншої служби. Можна помітити, що, незважаючи на те, що служба відрізняється, вона все ще включає концепцію запуску, налаштування, виконання та завершення. Виконання тут знову полягає в доступі до джерела даних із використанням деякого введення

користувача (у цьому випадку інтервал часу та конкретної будівлі), а потім виведення (обробленої) інформації.

3.5. Реалізація мікросервісу моніторингу і обміну даними на основі IoT

Впровадження цього мікросервісу моніторингу також покращує виконання вимог R1 , оскільки структура здатна контролювати виконання та реєстрацію пристроїв і R3a, оскільки формат введення/виведення кожного пристрою відомий і може бути перевірений. Діаграма активності цього мікросервісу зображена на рисунку 3.6, а також його опис.

1. Мікросервіс запускається періодично за замовчуванням, незалежно від розгорнутої програми.

2. Виявлення мережі - виявлено поточний стан мережі. На цьому етапі всі вузли сигналізують про своє існування вузлу і передбачається, що стан мережі постійно зберігається в фреймворку. Це актуально для наступного виконання мікросервісу.

2. Порівняти / оновити стан мережі - відмінності в мережі підтверджуються. Якщо немає змін, потік можна завершити.

3. Перевірка вимоги до програми - вимоги до апаратного та програмного забезпечення.

4. Можливість розгортання – виконується порівняння між вимогами програми та доступними ресурсами в мережі.

5. Оновлення вузлів — нова конфігурація розгортається в мережі.

6. Оновлення стану мережі. Решта кроків у мікросервісі гарантують, що окремі вузли оновлено, а поточний стан мережі та програми збережено.

Захист реалізовано, щоб переконатися, що не більше ніж один вузол у мережі може запускати потік моніторингу, знову ж таки, протокол мережевої синхронізації також має сприяти цій функції. Ми не можемо гарантувати R2 , оскільки це також залежить від зв’язку пристрою, але важливо, щоб служба

моніторингу була якомога меншою, щоб фаза перевірки та конфігурації часу виконання програми була зведена до мінімуму.

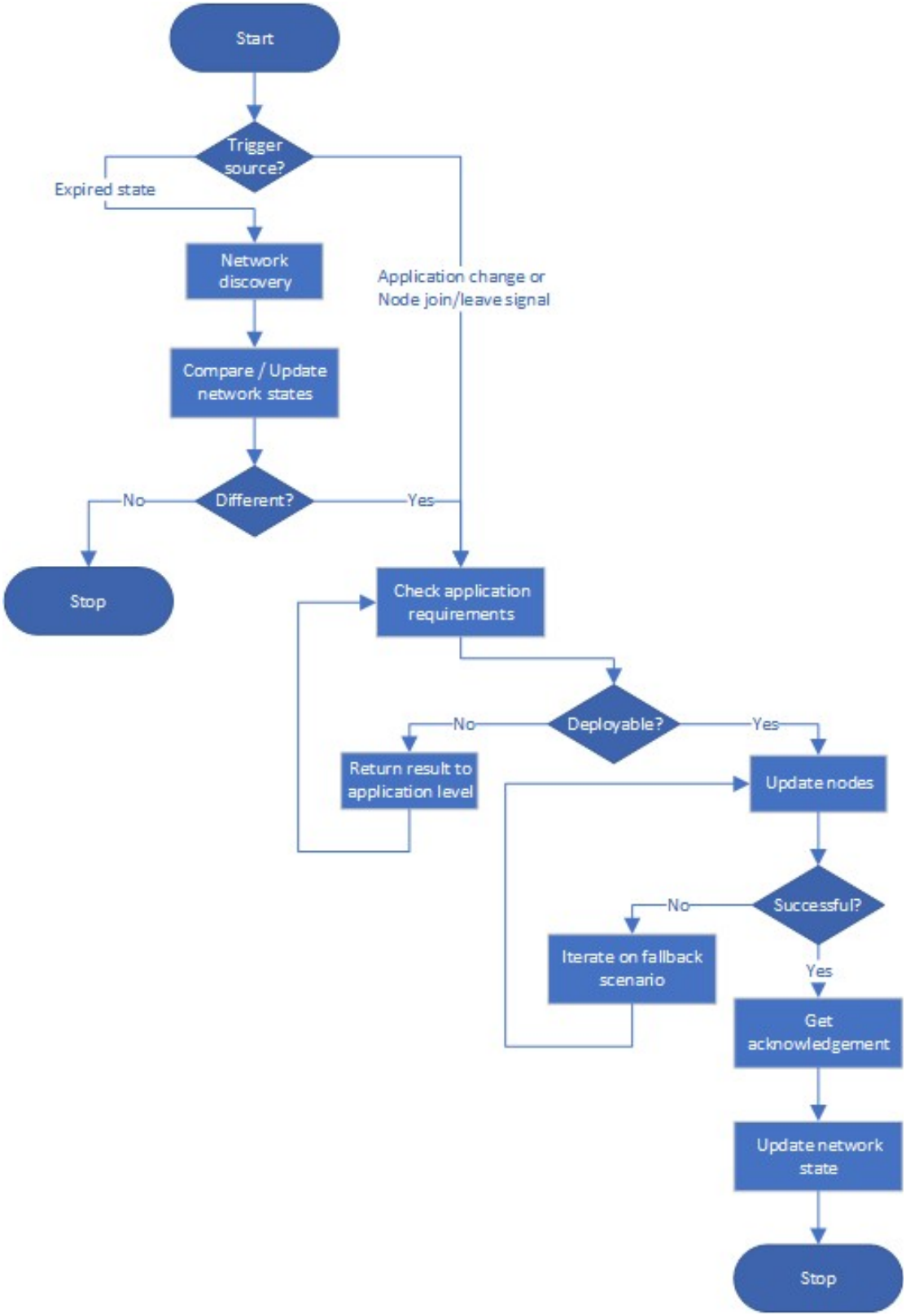

Рис. 3.6. Мікросервіс, відповідальний за моніторинг вузлів у мережі та розгортання застосунків на них

На рисунку 3.7 представлена діаграма класів для розробленої структури. Як результат аналізу, проведеного в даному розділі, основна характеристика архітектури фреймворку полягає в тому, що апаратне та програмне забезпечення відокремлюють свою основну характеристику, і вона включає основні принципи проектування, перелічені в даному розділі.

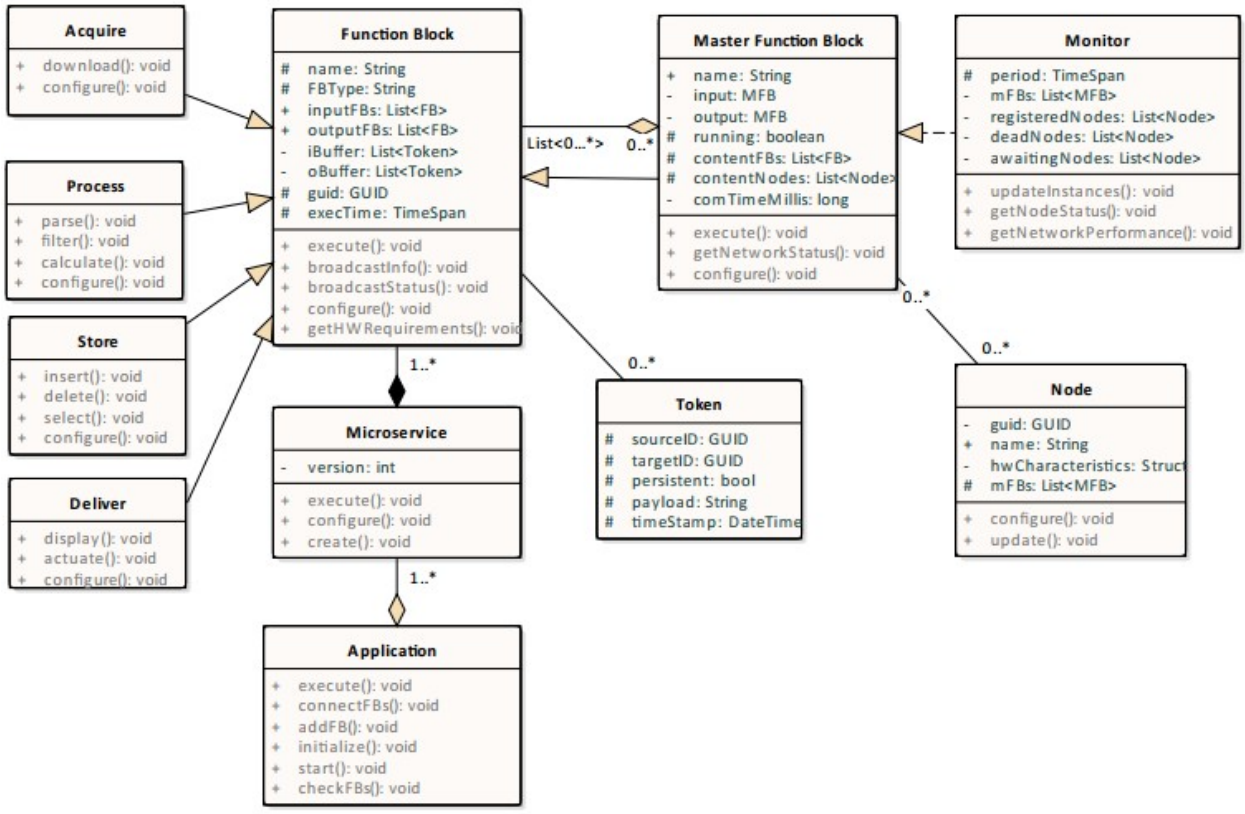

Рис. 3.7. Запропонована діаграма класів

Основу системи становить мікросервіс, який координує вузли всередині мережі. Як зазначено в S4, то вузли мають засоби сигналізації про свій статус (доступний / виконується). Є кілька підкомпонентів або функціональних блоків, які складають повне рішення для впровадження програми. 4 типи FB показані ліворуч на рисунку 3.7 і далі описані в цьому розділі. Ці функції також згадуються в S6.

Основною спільною характеристикою цих функціональних блоків є те, що вони тим чи іншим способом обробляють інформацію. Вони

розрізняються з точки зору можливих входів і виходів, а також доступних методів реалізації.

У лівій частині рисунка 3.8 визначено стандартні входи/виходи для функціонального блоку. Горизонтальні лінії представляють входи та виходи з або до інших функціональних блоків FB (S10) . Вертикальні лінії є частиною етапу налаштування програми. Щоб повною мірою скористатися цією функцією, програми повинні мати можливість компоновки (S2).

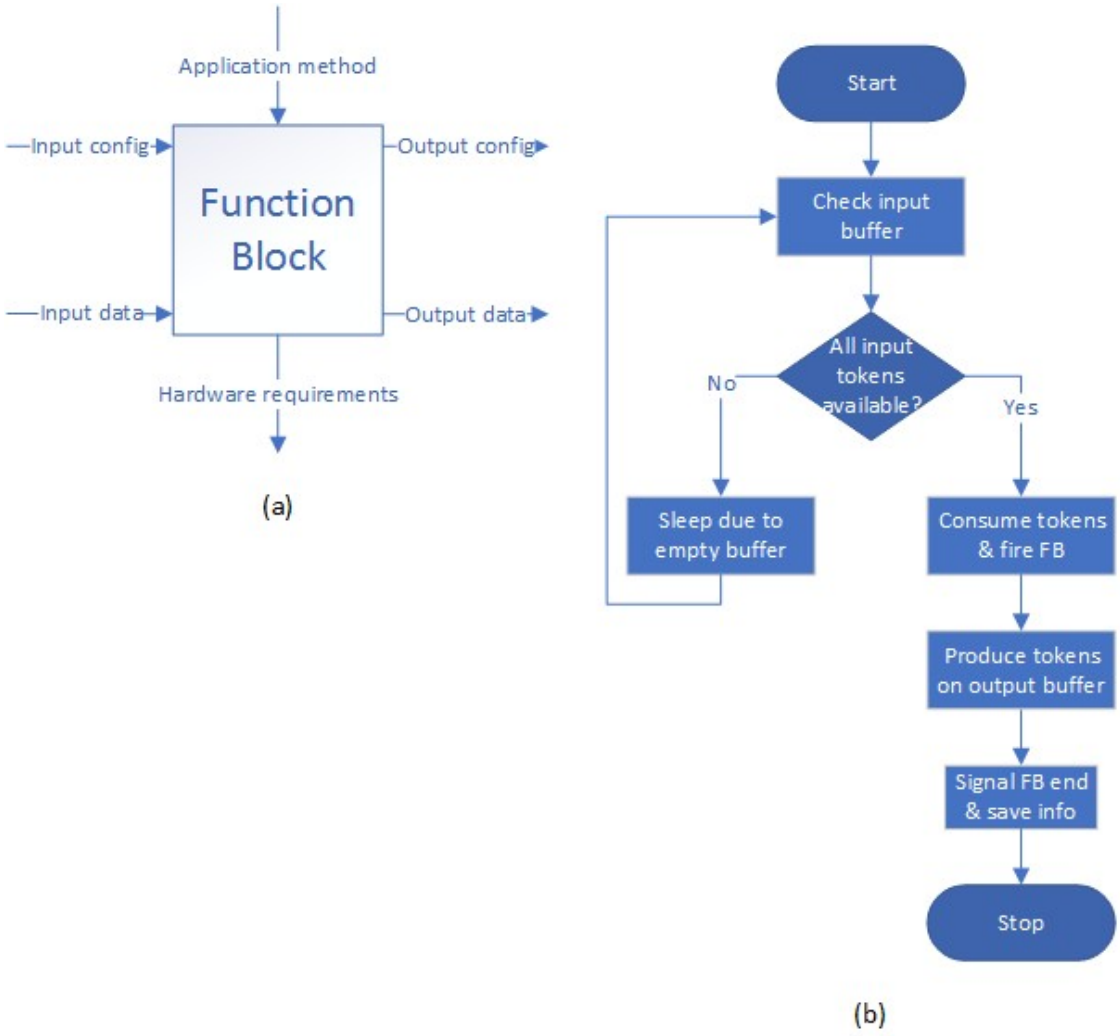

Рис. 3.8. Процеси введення/виведення даних для загальних функціональних блоків (a) та їх виконання (b)

Діаграма активності, показана на рисунку 3.8, описує стандартне виконання цього компонента. FB підкоряється базовому принципу потоку

даних споживання/вироблення одного токена на ребро під час виконання. Це також означає, що паралелізм можливий (S3).

Після переконфігурації програми, незалежно від їх типу, FB приписуються нові GUID, щоб гарантувати, що старі екземпляри FB відкидаються.

Завдяки використанню стандартизованих FB мікросервіс моніторингу може реалізовувати RPC, долаючи головний недолік архітектури мікросервісів і підтримуючи R1. Ця стандартизація також підтримує R2, оскільки ця система покращує точне відстеження продуктивності окремих FB, але це не може бути гарантовано, оскільки це також залежить від визначених QA програми S5.

Висновки до розділу

У цьому розділі були детально розглянуті етапи розробки сценарію використання, що дозволило визначити специфічні потреби та завдання, які вирішує запропонована система. Це включало врахування різноманітних можливих сценаріїв застосування IoT у контексті інтелектуальних мереж, що стало основою для подальшого проектування системи. Виконано узагальнення ключових результатів, отриманих в процесі розробки, проектування та реалізації системи обміну даними з використанням технологій Інтернету речей (IoT).

Проектування системи базувалося на основних принципах створення IoT-рішень, де були враховані особливості розподіленої архітектури та вимоги до безпеки і надійності передачі даних. Це дозволило забезпечити гнучкість системи та її здатність до масштабування в умовах зростаючих обсягів даних, що генеруються пристроями Інтернету речей у різних середовищах.

Значну увагу було приділено формулюванню функціональних та нефункціональних вимог до структури обміну даними. Функціональні

вимоги були зосереджені на забезпеченні надійного збору, передачі та обробки даних з пристроїв IoT, а також інтеграції цих даних у загальну мережеву інфраструктуру. Нефункціональні вимоги, такі як продуктивність, масштабованість та відмовостійкість, стали критично важливими для забезпечення стабільної роботи системи в умовах великих навантажень та динамічно змінних умов.

Запропонована концепція застосування IoT для обміну даними в інтелектуальних мережах дала змогу визначити основні принципи взаємодії між пристроями та забезпечення їх безперебійної роботи в умовах складних мережевих сценаріїв. Це підкреслює важливість стандартизації протоколів та інтеграції з існуючими мережевими рішеннями для досягнення сумісності та взаємодії між різними елементами системи.

ВИСНОВКИ

В магістерській роботі виконано імплементацію концептуального фреймворку Інтернету речей для обміну даними в інтелектуальних мережах. У процесі дослідження були розглянуті ключові компоненти та принципи роботи Інтернету речей, що створює базу для подальшого розуміння способів інтеграції IoT у розумні мережі. Було проаналізовано загальну архітектуру IoT, яка складається з декількох рівнів, що забезпечують зв'язок між фізичними пристроями, платформами обробки даних та користувацькими додатками. Це дозволило виявити основні елементи, які впливають на побудову ефективних IoT-систем.

Розгляд очікуваної ситуації та мети проекту дав змогу визначити ключові задачі, які стоять перед системами IoT в контексті їх використання в інтелектуальних мережах, зокрема забезпечення надійного і безперервного зв'язку між різноманітними пристроями та сенсорами. Також було виділено основні завдання для подальшого впровадження IoT у смарт-мережі, що включають підвищення рівня автоматизації та оптимізацію процесів управління ресурсами.

Дослідження особливостей комунікації в розумних мережах продемонструвало, що взаємодія між IoT-пристроями вимагає використання спеціалізованих протоколів для передачі даних, які забезпечують безпеку та ефективність передачі в умовах великої кількості підключень. В цьому контексті було проведено класифікацію існуючих протоколів IoT, що застосовуються в розумних мережах, таких як MQTT, CoAP та інших. Розгляд архітектурних особливостей і специфікацій цих протоколів допоміг краще зрозуміти їх можливості та обмеження.

Важливим аспектом є також аналіз перспектив використання IoT і розумних мереж, що підкреслює значний потенціал технологій IoT у розвитку різноманітних галузей, від енергетики до розумних будинків і міст. Інтеграція IoT із розумними мережами відкриває можливості для оптимізації

управління ресурсами, покращення екологічних показників та підвищення рівня комфорту для кінцевих користувачів.

В процесі реалізації мікросервісу для моніторингу та обміну даними на основі IoT були розроблені окремі модулі, що забезпечують збір даних, їх аналіз і передачу у відповідні компоненти системи. Така модульна структура дозволила створити гнучке рішення, яке може бути адаптоване до конкретних потреб користувачів та легко інтегрується в інтелектуальні мережі. Результати цього розділу демонструють ефективність обраного підходу та його здатність забезпечувати надійний обмін даними в умовах складної архітектури розподілених систем.

Таким чином, результати розділу підтверджують, що впровадження систем IoT в інтелектуальні мережі потребує ретельного врахування архітектури, протоколів і специфіки взаємодії між пристроями. Це забезпечує основу для подальшої розробки та імплементації ефективних IoT-рішень, спрямованих на підвищення продуктивності та функціональності розумних мереж.

ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Yuvraj Agarwal et al. “Occupancy-driven energy management for smart building automation”. In: Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building - BuildSys ’10. New York, New York, USA: ACM Press, 2010, p. 1. ISBN: 9781450304580. DOI: 10.1145/1878431.1878433. URL: http : / / portal . acm . org / citation . cfm ? doid = 1878431.
2. Debasis Bandyopadhyay and Jaydip Sen. “Internet of Things: Applications and Challenges in Technology and Standardization”. In:Wireless Pers Commun 58 (2011), pp. 49–69. DOI: 10.1007/s11277-011-0288-5. URL: https://link.springer.com/content/pdf/10.1007%7B%5C%%7D2Fs11277-011-0288-5.pdf.
3. Stefan Boschert and Roland Rosen. “Digital Twin—The Simulation Aspect”. In: Mechatronic Futures. Cham: Springer International Publishing, 2016, pp. 59 – 74. DOI: 10.1007/978-3-319-32156-1_5. URL: http://link.springer.com/10.1007/978-3-319-32156-1%7B%5C_%7D5.
4. Corporate Inc. Amazon. Amazon.com: Amazon Echo & Alexa Devices: Amazon Devices & Accessories. URL: https://www.amazon.com/Amazon-Echo- And-Alexa-Devices/b/?ie=UTF8%7B%5C&%7Dnode=9818047011%7B%5C&%7Dref%7B%5C_%7D=sv%7B%5C_%7Ddevicesubnav%7B%5C_%7D1
5. Corporate Inc. Google. Google Nest Hub - Smarthome-controller - Google Store. URL: https://store.google.com/product/google%7B%5C_%7Dnest.
6. Corporate Inc. Philips. How Philips Hue smart lighting works | Philips Hue. URL:https://www2.meethue.com/en-us/how-it-works.
7. Ivica Crnkovic. “Component-based software engineering ? new challenges in software development”. In: Software Focus 2.4 (2001), pp. 127–133. ISSN: 1529-7942. DOI: 10.1002/swf.45. URL: http://doi.wiley.com/10.1002/swf.45.

8. Pratikkumar Desai, Amit Sheth, and Pramod Anantharam. “Semantic Gateway as a Service Architecture for IoT Interoperability”. In: Proceedings – 2015 IEEE 3rd International Conference on Mobile Services, MS 2015. IEEE, June 2015, pp. 313–319. ISBN: 9781467372848. DOI: 10 . 1109 / MobServ . 2015 . 51. arXiv:1410.4977. URL: http://ieeexplore.ieee.org/document/7226706/.
9. Embitel. Story of Evolution of IoT | Adoption of internet of Things | Embitel. 2018. URL: https://www.embitel.com/blog/embedded-blog/unravelingthe-story- of- evolution- of- iot- and- its- rapid- adoption
10. Claudiu-George Ilincanu. “Towards Predictive Maintenance - Architecture for Automatic Integration of Field Devies in Industrial Automation”. MA thesis. Technical University Eindhoven, 2017, p. 70.
11. Yuna Jeong et al. “AVIoT: web-based interactive authoring and visualization of indoor internet of things”. In: IEEE Transactions on Consumer Electronics 61.3 (Aug. 2015), pp. 295–301. ISSN: 0098-3063. DOI: 10.1109/TCE.2015.7298088. URL: http://ieeexplore.ieee.org/document/7298088/.
12. Martin Keen et al. Patterns: Implementing an SOA Using an Enterprise Service Bus us. Tech. rep. URL: http://ck20.com/MQ/WBIMB/sg246346%20Implementing%20SOA%20using%20ESB.pdf.
13. Andrew S Tanenbaum Maarten and Van Steen. DISTRIBUTED SYSTEMS 121ritaci tple TAYAdiets Second Edition. Tech. rep. URL: http : / / cds . cern.ch/record/1056310/files/0132392275%7B%5C_%7DTOC.pdf.
14. Prateep Misra. Build a Scalable Platform for High-Performance IoT Applications. Tech. rep. 2016. URL: http://www.gartner.com/newsroom/id/3165317.
15. MuleSoft(C). The top 6 microservices patterns. URL: https://www.slideshare.net/AbhishekSood10/the-top-6-microservices-pattern-83814065.

16. Rahmani, A.M., Liljeberg, P., & Preden, J.S. (2022). A Survey on IoT-Enabled Smart Grids: Emerging Applications, Challenges, and Outlook. MDPI, Energies, 15(19), 6984.
17. Zhang, Y., Wang, Z., & Lin, Z. (2021). IoT-Enabled Smart Energy Grid: Applications and Challenges. IEEE Transactions on Industrial Informatics, 17(4), 1234-1245.
18. Sharma, S., & Lee, M. (2020). Communication Protocols for IoT-Based Smart Grid. Springer, Lecture Notes in Electrical Engineering, 499, 223-240.
19. Park, J., & Kim, H. (2023). An Efficient Interface for the Integration of IoT Devices with Smart Grids. MDPI, Sensors, 18(6), 332.
20. Williams, T., & Brown, R. (2022). Internet of Things (IoT) in Smart Grid: Technology, Security, and Future Directions. IEEE Xplore, Proceedings of the IEEE, 110(5), 675-690.
21. Bagdadee, A.H.; Zhang, L. Smart Grid: A Brief Assessment of the Smart Grid Technologies for Modern Power System. J. Eng. Technol. 2019, 8, 122–142.
22. Heirman, D. What makes Smart Grid-Smart—In addition, who is in the “game”? IEEE Electromagn. Compat. Mag. 2012, 1, 95–99.
23. Sakthivel, P.; Ganeshkumaran, S. Design of automatic power consumption control system using smart grid—A review. In World Conference on Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave); IEEE: Los Alamitos, CA, USA, 2016; pp. 1–4.
24. Dai, J.; Dong, M.; Ye, R.; Ma, A.; Yang, W. A review on electric vehicles and renewable energy synergies in smart grid. In Proceedings of the China International Conference on Electricity Distribution (CICED), Xi’an, China, 10–13 August 2016; pp. 1–4.
25. Qazi, A.; Hussain, F.; Rahim, N.A.; Hardaker, G.; Alghazzawi, D.; Shaban, K.; Haruna, K. Towards Sustainable Energy: A Systematic Review of

Renewable Energy Sources, Technologies, and Public Opinions. IEEE Access 2019, 7, 63837–63851.

26. Almehizia, A.A.; Al-Masri, H.M.K.; Ehsani, M. Integration of Renewable Energy Sources by Load Shifting and Utilizing Value Storage. IEEE Trans. Smart Grid 2019, 10, 4974–4984.
27. Zafar, R.; Mahmood, A.; Razzaq, S.; Ali, W.; Naeem, U.; Shehzad, K. Prosumer based energy management and sharing in smart grid. Renew. Sustain. Energy Rev. 2018, 82, 1675–1684.
28. Rehmani, M.H.; Reisslein, M.; Rachedi, A.; Erol-Kantarci, M.; Radenkovic, M. Integrating Renewable Energy Resources Into the Smart Grid: Recent Developments in Information and Communication Technologies. IEEE Trans. Ind. Inform. 2018, 14, 2814–2825.
29. Ghorbanian, M.; Dolatabadi, S.H.; Masjedi, M.; Siano, P. Communication in Smart Grids: A Comprehensive Review on the Existing and Future Communication and Information Infrastructures. IEEE Syst. J. 2019, 13, 4001–4014.
30. Zhang, Y.; Zhang, Y.; Mu, X.L.; Lei, X.H.; Li, F.; Zhou, J.; Xu, L.; Gao, Y.X.; Liu, X.Z. Development of IEC 61850 and Its Application. In Proceedings of the International Conference on Computer, Network Security and Communication Engineering (CNSCE), Bangkok, Thailand, 26–27 March 2017.
31. Englert, H.; Dawidczak, H. IEC 61850 substation to control center communication—Status and practical experiences from projects. In Proceedings of the 2009 IEEE Bucharest PowerTech, Bucharest, Romania, 28 June–2 July 2009; pp. 1–6.
32. Horalek, J.; Matyska, J.; Sobeslav, V. Communication protocols in substation automation and IEC 61850 based proposal. In Proceedings of the IEEE 14th International Symposium on Computational Intelligence and Informatics (CINTI), Budapest, Hungary, 19–21 November 2013; pp. 321–326.

33. Gupta, S. Communication in substation automation systems. Int. J. Res. Adv. Eng. Technol. 2016, 2, 54–58.
34. Ozansoy, C.; Zayegh, A.; Kalam, A. Communications for substation automation and integration. In Proceedings of the Australasian Universities Power Engineering Conference, Melborn, Australia, 29 September–2 October 2002.
35. Communication Networks and Systems for Power Utility Automation—Part 8-1: Specific Communication Service Mapping (SCSM)—Mappings to MMS (ISO 9506-1 and ISO 9506-2) and to ISO/IEC 8802-3, 2nd ed.; document IEC 61850-8-1; IEC: Geneva, Switzerland, 2011.
36. Communication Networks and Systems for Power Utility Automation—Part 7-420: Basic Communication Structure—Distributed Energy Resources Logical Nodes, 1st ed.; document IEC 61850-7-420; IEC: Geneva, Switzerland, 2009.
37. Communication Networks and Systems for Power Utility Automation—Part 90-7: Object Models for Power Converters in Distributed Energy Resources (DER) Systems, 1st ed.; document IEC 61850-90-7; IEC: Geneva, Switzerland, 2013.
38. Zaballos, A.; Vallejo, A.; Selga, J.M. Heterogeneous communication architecture for the smart grid. IEEE Netw. 2011, 25, 30–37.
39. Eriksson, M.; Armendariz, M.; Vasilenko, O.O.; Saleem, A.; Nordström, L. Multiagent-Based Distribution Automation Solution for Self-Healing Grids. IEEE Trans. Ind. Electron. 2015, 62, 2620–2628.
40. Saleem, Y.; Crespi, N.; Rehmani, M.H.; Copeland, R. Internet of Things-Aided Smart Grid: Technologies, Architectures, Applications, Prototypes, and Future Research Directions. IEEE Access 2019, 7, 62962–63003.
41. Communication Networks and Systems for Power Utility Automation—Part 8-2: Specific Communication Service Mapping (SCSM)—Mapping to Extensible Messaging Presence Protocol (XMPP), 1st ed.; document IEC 61850-8-2; IEC: Geneva, Switzerland, 2018.

42. Al-Ali, A. Role of internet of things in the smart grid technology. J. Comput. Commun. 2015, 3, 229. [Green Version]
43. Kaur, M.; Kalra, S. A review on IOT based smart grid. Int. J. Energy Inf. Commun. 2016, 7, 11–22.
44. Viswanath, S.K.; Yuen, C.; Tushar, W.; Li, W.; Wen, C.; Hu, K.; Chen, C.; Liu, X. System design of the internet of things for residential smart grid. IEEE Wirel. Commun. 2016, 23, 90–98. [Green Version]
45. Dalipi, F.; Yayilgan, S.Y. Security and Privacy Considerations for IoT Application on Smart Grids: Survey and Research Challenges. In Proceedings of the IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW), Vienna, Austria, 22–24 August 2016; pp. 63–68.
46. Sakhnini, J.; Karimipour, H.; Dehghantanha, A.; Parizi, R.M.; Srivastava, G. Security aspects of Internet of Things aided smart grids: A bibliometric survey. Internet Things 2019, 16, 100111.
47. Bikmetov, R.; Raja, M.Y.A.; Sane, T.U. Infrastructure and applications of Internet of Things in smart grids: A survey. In Proceedings of the North American Power Symposium, Morgantown, WV USA, 17–19 Septrember 2017; pp. 1–6.
48. Jiang, A.; Yuan, H.; Li, D.; Tian, J. Key technologies of ubiquitous power Internet of Things-aided smart grid. J. Renew. Sustain. Energy 2019, 11, 062702. [Green Version]
49. Ghasempour, A. Internet of Things in Smart Grid: Architecture, Applications, Services, Key Technologies, and Challenges. Invention 2019, 4, 22. [Green Version]
50. Collier, S.E. The Emerging Enernet: Convergence of the Smart Grid with the Internet of Things. IEEE Ind. Appl. Mag. 2017, 23, 12–16.
51. Reka, S.S.; Dragicevic, T. Future effectual role of energy delivery: A comprehensive review of Internet of Things and smart grid. Renew. Sustain. Energy Rev. 2018, 91, 90–108.

52. Bedi, G.; Venayagamoorthy, G.K.; Singh, R.; Brooks, R.R.; Wang, K. Review of Internet of Things (IoT) in Electric Power and Energy Systems. IEEE Internet Things J. 2018, 5, 847–870.
53. Bahashwan, A.A.O.; Manickam, S. A Brief Review of Messaging Protocol Standards for Internet of Things (IoT). J. Cyber Secur. Mobil. 2019, 8, 1–14.
54. Leila Fatmasari Rahman, Tanir Ozcelebi, and Johan J. Lukkien. “Choosing Your IoT Programming Framework: Architectural Aspects”. In: 2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud). IEEE, Aug. 2016, pp. 293–300. ISBN: 978-1-5090-4052-0. DOI: 10.1109/FiCloud.2016. – URL: http://ieeexplore.ieee.org/document/ 7575/.