НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Навчально-науковий Інститут телекомунікаційних систем

Кафедра електронних комунікацій та інтернету речей

«До захисту допущено»
ВО завідувача кафедри
_____ Галина СОЗОННИК
«___»_____20__ р.

Дипломна робота

на здобуття ступеня бакалавра

зі спеціальності 172 Телекомунікації та радіотехніка

на тему: «Протоколи рівня застосунків в мережах IoT та їх застосування»

Виконала:
студентка IV курсу, групи ТС-01
Дідковська Наталія Андріївна _____

Керівник:
Доцент каф. ЕКІР к.т.н., доц.
Григоренко О.Г. _____

Рецензент:
Доцент каф. Інформаційної безпеки НН ФТІ, доц., к.т.н.
Репа Ф.М. _____

Засвідчую, що у цій дипломній роботі
немає запозичень з праць інших авторів
без відповідних посилань.
Студентка _____

Київ – 2024 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Навчально-науковий Інститут телекомунікаційних систем
Кафедра електронних комунікацій та інтернету речей

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 172 Телекомунікації та радіотехніка

Освітня програма – «Системи електронних комунікацій та інтернету речей»

ЗАТВЕРДЖУЮ
ВО завідувача кафедри
_____ Галина СОЗОННИК
«___»_____20__ р.

ЗАВДАННЯ
на дипломну роботу студенту
Дідковській Наталії Андріївні

1. Тема роботи «Протоколи рівня застосунків в мережах IoT та їх застосування», керівник роботи Григоренко Олена Григорівна, доцент, кандидат технічних наук, затверджені наказом по університету від «22» травня 2024 р. № 2064-С

2. Термін подання студентом роботи 14.06.2024

3. Вихідні дані до роботи: наукові статті, підручники за темою технології IoT.

4. Зміст роботи:

1. Особливості мереж електронних комунікацій та IoT
2. Транспортні мережі та мережі доступу
3. Архітектура мережі IoT
4. Протоколи рівня застосунків в мережах IoT
5. Використання протоколів рівня застосунків

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо): електронна презентація PowerPoint:

Слайд № 1 Титульний слайд

Слайд № 2 Мета та актуальність роботи

Слайд № 3 Транспортні мережі та мережі доступу

Слайд № 4 Місце мережі IoT в сучасній телекомунікаційній інфраструктурі

Слайд № 5 Архітектура мережі IoT

Слайд № 6 Протокол MQTT

Слайд № 7 Протокол DDS

Слайд № 8 Протокол CoAP

Слайд № 9 Протокол AMQP

Слайд № 10 Протокол HTTP

Слайд № 11 Аналіз та порівняння протоколів

Слайд № 12,13 Приклади використання протоколів рівня застосунків

Слайд № 14 Хмарна обробка

Слайд № 15 Взаємодія з протоколами інших рівнів

Слайд № 16 Висновки

6. Дата видачі завдання 06.11.2023

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1.	Вивчення літератури та стандартів з дипломної роботи	08.01.24-31.01.24	Виконано
2.	Написання змісту та вступу до роботи	01.02.24-05.02.24	Виконано
3.	Розгляд особливостей мереж електронних комунікацій та IoT	12.02.24-01.03.24	Виконано
4.	Розгляд протоколів рівня застосунків в мережах IoT	18.03.24-26.04.24	Виконано
5.	Розгляд використань протоколів рівня застосунків	06.05.24-24.05.24	Виконано
6.	Написання висновків	27.05.24-31.05.24	Виконано
7.	Оформлення роботи	03.06.24-07.07.24	Виконано
8.	Розробка доповіді та презентації	10.06.24-12.06.24	Виконано

Студент Наталія ДІДКОВСЬКА

Керівник роботи Олена ГРИГОРЕНКО

РЕФЕРАТ

Дипломна робота обсягом 71 сторінки складається з переліку скорочень, вступу, трьох розділів, загальних висновків та переліку посилань на 40 джерел. Також містить 34 рисунки та 1 таблицю .

У першому розділі проведено аналіз транспортної мережі та мережі доступу. Визначено місце мережі IoT в сучасній телекомунікаційній інфраструктурі. А також розглянуто та проаналізовано архітектуру мережі IoT: рівень сенсорів та сенсорних мереж, рівень шлюзів та мереж, сервісний рівень та рівень додатків.

У другому розділі були розглянуті та проаналізовані протоколи рівня застосунків: MQTT, DDS, CoAP, AMQP та HTTP. Також був проведений порівняльний аналіз у вигляді таблиці.

У третьому розділі було розглянуто перспективи розвитку даних протоколів та різні галузі їх використання. На останок було проаналізовано хмарну обробку та взаємодію з протоколами інших рівнів.

Ключові слова: IoT, Інтернет речей, протокол, протоколи рівня застосунків, мережа, MQTT, DDS, CoAP, AMQP, HTTP, IIoT.

ABSTRACT

The 71-page thesis consists of a list of abbreviations, an introduction, three chapters, general conclusions, and a list of 40 references. It also contains 34 figures and 1 table.

The first chapter analyzes the transportation and access networks. The place of the IoT network in the modern telecommunications infrastructure is determined. Also, the architecture of the IoT network is considered and analyzed: the level of sensors and sensor networks, the level of gateways and networks, the service level and the application level.

In the second section, the application layer protocols were reviewed and analyzed: MQTT, DDS, CoAP, AMQP, and HTTP. A comparative analysis in the form of a table was also conducted.

In the third section, the prospects for the development of these protocols and various areas of their use were considered. Finally, cloud processing and interaction with other protocols were analyzed.

Keywords: IoT, Internet of Things, protocol, application layer protocols, network, MQTT, DDS, CoAP, AMQP, HTTP, IIoT.

Зміст

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

СФТТ	сегментом формування транзитного трафіку
LAN	Local Area Network – Локальна обчислювальна мережа
MAN	Metropolitan Area Network – Міська обчислювальна мережа
WAN	Wide Area Network – Глобальна обчислювальна мережа
OSI	Open Systems Interconnection (Модель взаємодії відкритих систем)
ISO	International Organization for Standardization – Міжнародна організація зі стандартизації)
FTTH	Fiber to the Home – Оптоволокно до дому
Wi-Fi	Wireless Fidelity
WiMAX	Worldwide Interoperability for Microwave Access – Світова взаємодія для мікрохвильового доступу
HFC	Hybrid Fiber-Coaxial – Гібридна оптико-коаксіальна мережа
NT	Network Termination – Мережеве закінчення
TE	Termination Equipment – Кінцеве обладнання
NU	Network Unit – Мережевий блок
DN	Distribution Node – Розподільчий вузол
ITU-T	International Telecommunication Union - Telecommunication Standardization Sector – Міжнародний союз електрозв'язку - Сектор стандартизації електрозв'язку
ANL	Access Network Line – Лінія мережі доступу
SL	Sabscriber Line – Абонентська лінія
DN	Distribution Network – Розподільча мережа
NDN	Network Distribution Node – Мережевий розподільчий вузол
BN	Backbone Network – Магістральна/транспортна мережа
PAN	(Personal Area Network) – Персональна мережа

GSM	Global System for Mobile Communications – Глобальна система мобільного зв'язку
UWB	Ultra-Wide Band – Надширокосмуговий зв'язок
GPRS	General Packet Radio Service – Служба пакетного радіозв'язку загального користування
LTE	Long-Term Evolution – Довгостроковий розвиток
WSN	Wireless Sensor Network – Бездротові сенсорні мережі
M2M	Machine-to-Machine – Міжмашинна взаємодія
OSS	Operation Support System – Підтримка операційної діяльності
BSS	Business Support System – Підтримка бізнес діяльності
BRM	Business Rule Management – Управління бізнес-правилами
BPM	Business Process Management – Управління бізнес-процесами
WoT	WEB of Things – Веб Речей
URI	Uniform Resource Identifier – Уніфікований ідентифікатор ресурсу
HTTP	Hypertext Transfer Protocol – Протокол передачі гіпертексту
REST	Representational State Transfer – Представлення переходу стану
MQTT	Message Queuing Telemetry Transport – Протокол передачі телеметричних повідомлень через чергу
QoS	Quality of Service – Рівень якості обслуговування
DDS	Data Distribution Service – Служба розподілу даних
CoAP	Constrained Application Protocol – Протокол обмеженого застосування
IIoT	Industrial Internet of Things – Промисловий Інтернет речей
TCP	Transport Control Protocol – Протокол керування передачею даних
IP	Internet Protocol – Протокол Інтернету
BLE	Bluetooth Low Energy – Bluetooth з низьким енергоспоживанням
DTLS	Datagram Transport Layer Security – Протокол забезпечення

	безпеки транспортного рівня датаграм
PSK	Pre-Shared Key – Попередній спільний ключ
RPK	Raw Public Key – Необроблений відкритий ключ
DoS	Denial of Service – Відмова в обслуговуванні
MAC	Media Access Control – Контроль доступу до середовища
AWS	Amazon Web Services
URI	Uniform Resource Identifier - Уніфікований ідентифікатор ресурсу
URL	Uniform Resource Locator - Універсальний локатор ресурсу
IPv4/6	Internet Protocol version 4/6 – Інтернет-протокол версії 4

ВСТУП

У сучасному світі технологічний розвиток набуває все більшого значення, яскравим прикладом являється Інтернет речей (IoT), що стає невід'ємною частиною повсякденного життя людей. IoT включає широкий спектр пристроїв, які можуть взаємодіяти між собою, обмінюватися даними та виконувати різноманітні завдання без втручання людини. Така взаємодія можлива завдяки надійним та ефективним протоколам зв’язку, зокрема протоколам рівня застосунків, які грають ключову роль у забезпеченні функціонування IoT мереж.

Протоколи рівня застосунків в мережах Інтернету речей відповідають за передачу даних між пристроями, управління та контролю, а також забезпечення безпеки та надійності комунікацій. Вони визначають правила, за якими взаємодіють пристрої один з одним, формати повідомлень та механізми обробки даних. Серед найпоширеніших протоколів рівня застосунків в IoT можна виділити MQTT, DDS, CoAP, AMQP та HTTP. Кожен з них має свої унікальні характеристики та підходить для різних типів застосувань.

Актуальність дослідження полягає в тому, що зростаюча кількість IoT пристроїв і застосувань вимагає ефективних рішень для обміну даними, що в свою чергу потребує глибокого розуміння протоколів, які використовуються на рівні застосунків. Від правильного вибору протоколу залежить ефективність, надійність та безпека всієї системи Інтернету речей. Тому важливо дослідити, які протоколи найкраще підходять для певних умов застосування, а також їхні переваги та недоліки. Не слід забувати і про інноваційні підходи, що можуть бути застосовані для їх покращення.

Метою даної дипломної роботи є аналіз протоколів рівня застосунків, які використовуються в мережах IoT, а також дослідження їх характеристик та виявлення областей їх застосування. Робота включає теоретичний огляд

основних протоколів, безпеки та ефективності, а також приклади реальних застосувань.

Об’єктом дослідження є мережі IoT та протоколи, які в них застосовуються.

Предметом дослідження є протоколи рівня застосунків, які використовуються в мережах Інтернету речей. А також вивчення специфікацій, функціональних можливостей та областей застосування таких протоколів, як MQTT, DDS, CoAP, AMQP та HTTP.

Наукова новизна полягає в тому, що дане дослідження сприятиме глибшому розумінню протоколів рівня застосунків в мережах Інтернету речей, а також допоможе розробникам та інженерам обирати найкращі рішення для конкретних проєктів, що сприятиме подальшому розвитку IoT та інтеграції цих технологій у різні сфери життя.

Для досягнення мети даної роботи потрібно вирішити наступні завдання:

1. Особливості мереж електронних комунікацій та IoT
2. Аналіз протоколів рівня застосунків в мережах IoT
3. Використання протоколів рівня застосунків

1 ОСОБЛИВОСТІ МЕРЕЖ ЕЛЕКТРОННИХ КОМУНІКАЦІЙ ТА IOT

1.1 Транспортні мережі та мережі доступу

«Транспортна мережа (Transport Network, Transmission Media) – це сегмент з високим ступенем концентрації трафіку, за допомогою якого здійснюється інформаційний обмін між сегментами з більш повільним трафіком і в якому транспортне середовище для передавання будь-якого типу інформації забезпечується використанням єдиних технологічних принципів і встановлених стандартів з надання ширини смуги пропускання» [1].

Звідси випливає, що основним завданням транспортної мережі є перерозподіл трафіку між усіма поєднаними елементами мережі. Дана мережа є сегментом формування транзитного трафіку (СФТТ). Варто зазначити, що такий сегмент може бути наявним у мережах, які охоплюють різні території наприклад: LAN (Local Area Network), MAN (Metropolitan Area Network), WAN (Wide Area Network) [1].

Згідно рисунку 1.1 побудова транспортних мереж базується на принципах побудови сегментів фізичного, канального та мережевого рівнів моделі OSI/ISO. Оскільки для сегментів LAN, MAN та WAN розроблено спочатку різні телекомунікаційні технології, то й елементи обладнання, які використовують мають певні відмінності [1].

Мережа доступу (Access Network) – це сегмент телекомунікаційної мережі, у якому формуються інформаційні потоки, які спрямовані в транспортну мережу.

З одного боку мережі доступу та транспортна мережа спільно вирішують завдання реалізації транспортної функції з перенесення інформації з кінця в кінець, але з іншого боку варто зазначити, що телекомунікаційні технології, які використовуються в них різняться [1].

З’єднання цих мереж здійснюється у вузлах доступу до транспортної мережі [1].

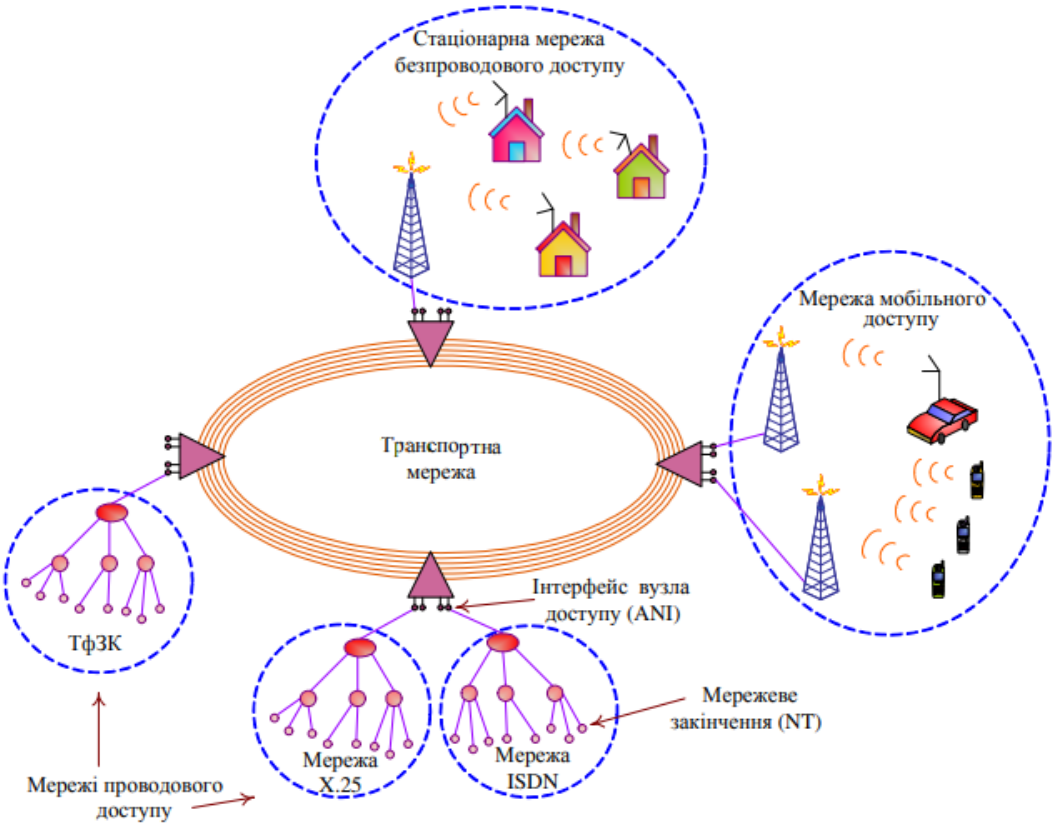

Рисунок 1.1 – Транспортна мережа та мережі доступу [1]

Мережі доступу узагальнено поділяються на:

- мережі проводового доступу (Fiber to the Home, Ethernet);
- бездротові мережі доступу (Wi-Fi, WiMAX, Bluetooth);
- гібридні мережі доступу (Hybrid Fiber-Coaxial).

«Мережі доступу, у загальному випадку, мають багаторівневу архітектуру, що включає вузли рівнів доступу, розподілу і ядра.

Опорні вузли мереж абонентського доступу формують рівень доступу.

Вузли рівня розподілу забезпечують агрегацію інформаційних потоків, що надходять від опорних вузлів абонентського доступу, і магістралями направляють агреговані потоки у вузли доступу до транспортної мережі. У вузлі доступу до транспортної мережі відбувається концентрація всіх інформаційних потоків від приєднаних вузлів рівня розподілу.

Вузол доступу до транспортної мережі, таким чином, переміщується на рівень ядра в мережі доступу. Якщо територіальна протяжність є значною, мережа доступу може розглядатися як самостійний сегмент MAN.

Розподільчою мережею (Distribution Network) називають сегмент телекомунікаційної мережі, за допомогою якого концентрований потік, який надходить з транспортної мережі, перерозподіляється та надходить до споживачів. На практиці функції мережі доступу та розподільчої мережі часто поєднуються в одному сегменті» [1].

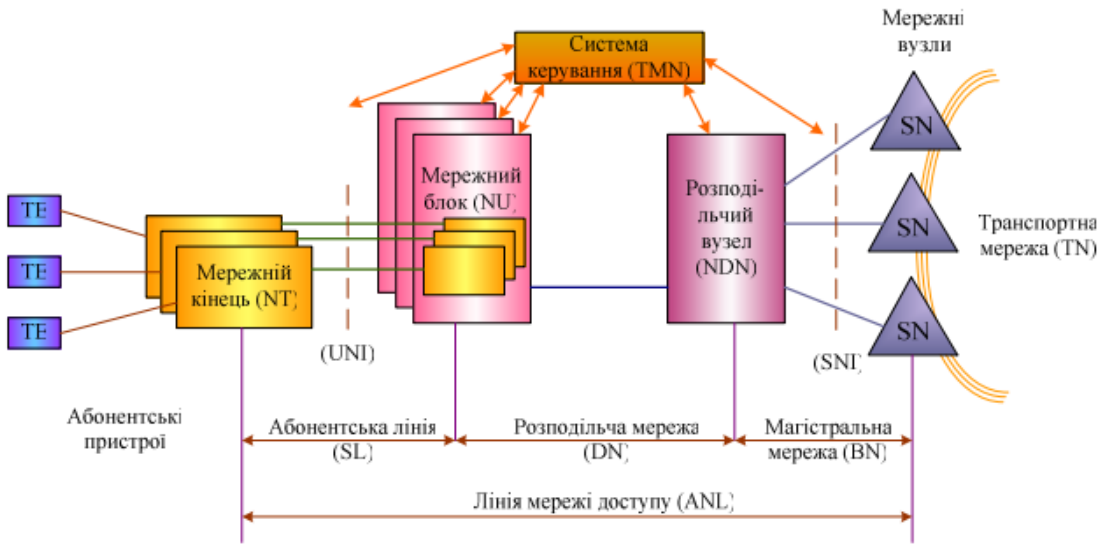

Рисунок 1.2 – Модель мережі доступу [1]

На рисунку 1.2 зображена мережа доступу, що являє собою сукупність абонентських ліній та обладнання місцевої мережі, які забезпечують доступ абонентських терміналів до транспортної мережі та місцевий зв'язок без виходу в транспортну мережу [1].

«Мережеве закінчення NT (Network Termination) дає змогу під’єднувати один або декілька користувацьких терміналів TE (Termination Equipment).

Мережевий блок NU (Network Unit) забезпечує первинний доступ через мультиплексування й концентрацію трафіку та каналів, а розподільчий

вузол DN (Distribution Node) – доступ абонентських пристроїв TE до сервісних вузлах SN.

У даній узагальненій моделі мережі доступу ITU-T уперше вводиться поняття «лінія мережі доступу» (Access Network Line, ANL). Це лінія, яка з'єднує мережеве закінчення NT з сервісним вузлом SN і проходить через усю мережу доступу. Вона може бути утворена фізичним ланцюгом (колом), каналом (аналоговим або цифровим), складовим каналом, віртуальним каналом або декількома каналами для однакових або різних послуг.

ANL проходить через абонентську лінію SL (Sabscriber Line), інтерфейс UNI, мережевий блок NU, розподільчу мережу DN (Distribution Network), мережевий розподільчий вузол NDN (Network Distribution Node) та магістральну (транспортну) мережу BN (Backbone Network)» [1].

1.2 Місце мережі IoT в сучасній телекомунікаційній інфраструктурі

Відомо, що Інтернет речей почав займати важливе місце в сучасній телекомунікаційній інфраструктурі, адже дана мережа дозволяє забезпечити зв’язок між різними пристроями, збирати та аналізувати всі дані в режимі реального часу [2;4].

Ось кілька аспектів, які підкреслюють значення IoT в сучасних телекомунікаціях:

По-перше, Інтернет речей дозволяє підключати величезну кількість пристроїв до мережі, створюючи масштабні системи для збору, передачі та аналізу даних. Завдяки IoT, можливо підключити пристрої в різних географічних розташуваннях, забезпечуючи глобальну взаємодію [2;4].

По-друге, мережі IoT сприяють створенню нових послуг, таких як розумні міста, розумні будинки, розумні транспортні системи та промисловий IoT. Телекомунікаційні компанії можуть пропонувати нові

сервіси на основі IoT, такі як віддалений моніторинг, управління енергоспоживанням та підвищення безпеки [2;4].

По-третє, інтеграція IoT допомагає оптимізувати існуючі телекомунікаційні мережі, покращуючи їх ефективність та знижуючи витрати. IoT дозволяє більш точно прогнозувати навантаження на мережі, що сприяє кращому плануванню ресурсів [2;4].

По-четверте, IoT сприяє впровадженню нових технологій, таких як 5G, які забезпечують більш високу пропускну здатність та знижену затримку. Взаємодія IoT та 5G створює умови для розвитку нових рішень, що потребують швидкої та надійної передачі даних [2;4].

По-п’яте, мережі IoT допомагають підвищити рівень безпеки в телекомунікаціях, забезпечуючи додатковий контроль за підключеними пристроями. Інтеграція IoT дозволяє впроваджувати більш розумні системи управління мережами, включаючи автоматизацію та самостійне виявлення проблем [2;4].

Пристрої мереж Інтернету речей підключаються до мереж доступу з використанням технологій Wi-Fi, Bluetooth, Zigbee Network, ZWawe Network далі цей трафік передається транспортною мережею. На рисунку 1.3 наведено місце мереж IoT в загальній телекомунікаційній інфраструктурі.

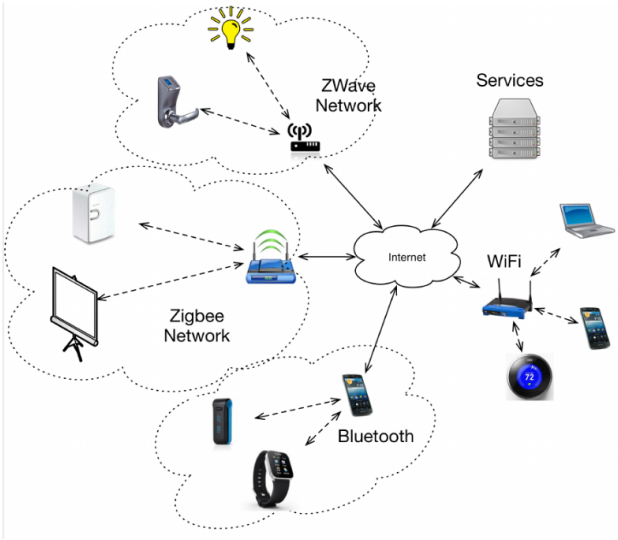

Рисунок 1.3 – Сучасний стан мережевої архітектури IoT [5]

1.3 Архітектура мережі IoT

Інтернет речей концептуально належить до мереж наступного покоління, тому його архітектура багато в чому схожа на відому чотиришарову архітектуру NGN. IoT складається з набору різних інфокомунікаційних технологій, що забезпечують функціонування Інтернету речей. Його ж архітектура показує, як ці технології пов'язані одна з одною. Архітектура IoT складається з кількох основних шарів (згідно рисунку 1.4), кожен з яких відіграє важливу роль у забезпеченні збирання, передачі, обробки та зберігання даних, а також надання кінцевих додатків та сервісів [3; 4].

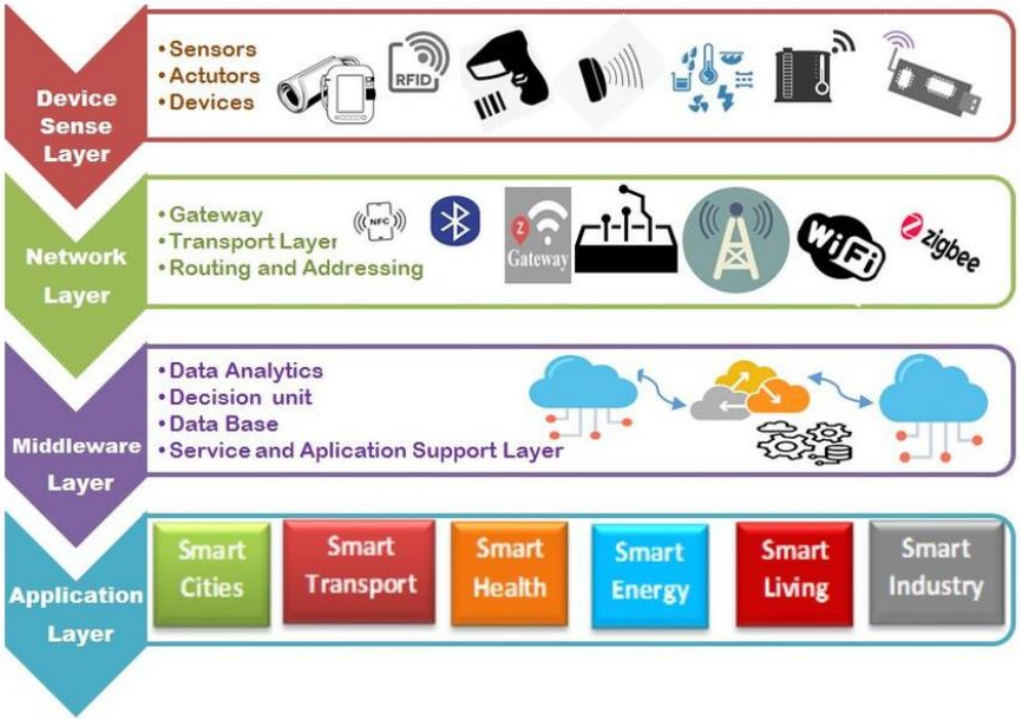

Рисунок 1.4 – Архітектура мережі IoT [6]

1.3.1 Рівень сенсорів та сенсорних мереж

Найнижчий рівень архітектури IoT складається з «розумних» об'єктів, інтегрованих з сенсорами або датчиками. Сенсори реалізують поєднання

фізичного та віртуального світів, забезпечуючи при цьому збирання та обробку інформації в режимі реального часу [3].

Існують різні типи сенсорів: сенсори навколишнього середовища, сенсори для медичних цілей, тобто для тіла людини, для побутової техніки та для транспортних засобів. Сенсори можуть мати як велику так і невелику пам'ять, дозволяючі записувати деяку кількість результатів вимірювань. Сенсор вимірює фізичні параметри контрольованого об'єкта або явища та перетворює їх на сигнал, який приймається відповідним пристроєм [3].

Більшість сенсорів вимагає з'єднання із шлюзом, який реалізується з використанням локальної обчислювальної мережі LAN (Local Area Network): Ethernet та Wi-Fi або персональної мережі PAN (Personal Area Network), таких як ZigBee, Bluetooth та UWB, (Ultra-Wide Band) бездротового зв'язку на малих відстанях [3].

Для сенсорів, які не потребують підключення до шлюзу, зв'язок із серверами може надаватися за допомогою глобальних бездротових мереж WAN: GSM (Global System for Mobile Communications), GPRS (General Packet Radio Service) та LTE (Long-Term Evolution). Сенсори, які характеризуються низьким енергоспоживанням та низькою швидкістю передачі даних, утворюють бездротові сенсорні мережі WSN (Wireless Sensor Network) [3].

Рисунок 1.5 – Розумні сенсори [7]

1.3.2 Рівень шлюзів та мереж

Великий обсяг даних, що створюється на першому рівні IoT різними видами сенсорів, вимагає надійної та високопродуктивного транспортного середовища, яке буде створене у вигляді провідної або бездротової мережної інфраструктури [3].

Існуючі мережі зв'язку, що використовують різні протоколи, можуть бути використані для підтримки між машинних комунікацій M2M та їх додатків. Цей рівень складається з мережевої інфраструктури, що створюється шляхом інтеграції мереж різних технологій та протоколів доступу у єдину мережеву платформу в інтересах реалізації широкого спектру послуг та додатків в IoT. Ці мережі повинні забезпечувати необхідні значення якості передачі інформації - затримку, пропускну спроможність та безпеку [3].

Шлюзи діють як маршрутизатори у цій архітектурі, адже вони керують мережею, транслюють протоколи зв’язку на сторону глобальної мережі та виконують команди, які надходять із локального сервера чи хмари. Проте

замість того, щоб покладатися виключно на Wi-Fi, вони можуть створити взаємопов’язану, малопотужну LAN за допомогою інших протоколів, таких як BLE, ZigBee, Z-Wave та Sub 1GHz [7].

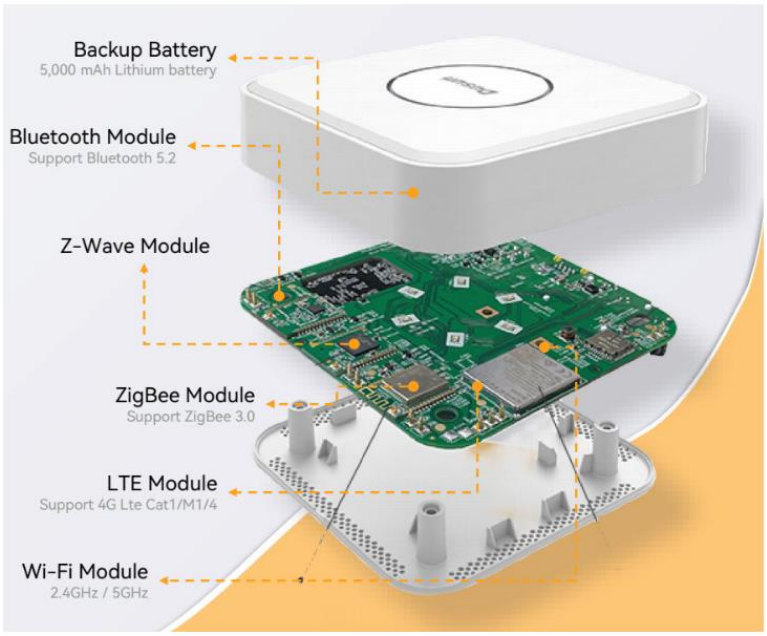

Рисунок 1.6 – Мережевий шар [7]

1.3.3 Сервісний рівень

«Сервісний рівень містить набір інформаційних послуг, покликаних автоматизувати технологічні та бізнес операції в IoT, а саме: підтримка операційної та бізнес діяльності (OSS/BSS, Operation Support System/Business Support System), різна аналітична обробка інформації (статистичний, інтелектуальний аналіз даних та текстів, прогностична аналітика та ін.), зберігання даних, забезпечення інформаційної безпеки, управління бізнес-правилами (BRM, Business Rule Management), управління бізнес-процесами (BPM, Business Process Management) та ін.» [3].

Рисунок 1.7 – Сервісний рівень [8]

1.3.4 Рівень додатків

«На четвертому рівні архітектури IoT існують різні типи додатків для відповідних промислових секторів та сфер діяльності (енергетика, транспорт, торгівля, медицина, освіта та ін.). Складовою частиною Інтернету речей є WEB of Things (WoT), який забезпечує взаємодію різних інтелектуальних об'єктів («речей») з використанням стандартів та механізмів Інтернету, таких як уніфікований ідентифікатор ресурсу URI (Uniform Resource Identifier), протокол передачі гіпертексту HTTP (HyperText Transfer Protocol), стиль побудови архітектури розподіленого додатку REST (Representational State Transfer) та ін. Фактично WoT передбачає реалізацію концепції IoT на прикладному рівні з використанням існуючих архітектурних рішень, орієнтованих на розробку web-додатків. Іншими словами, дані з розумних речей або управління ними мають бути доступним через WWW-сторінки» [3].

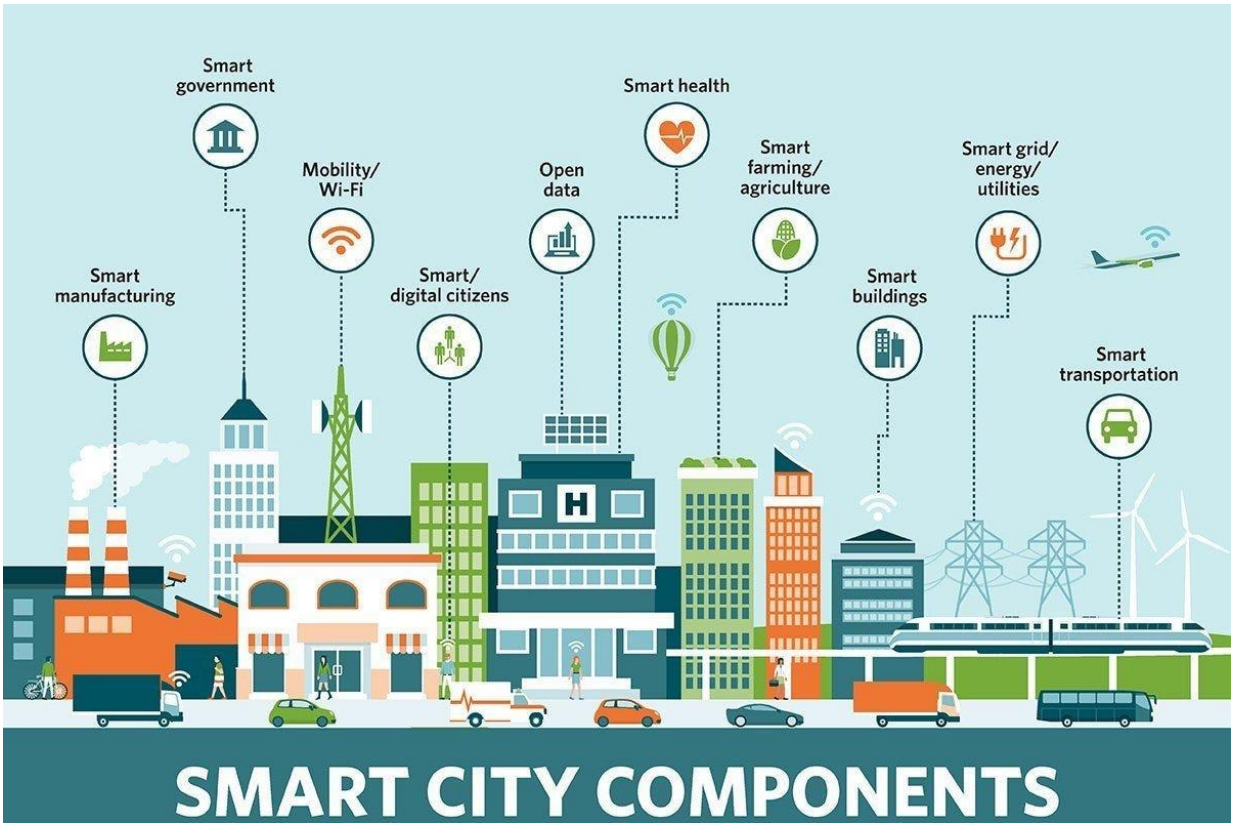

Рисунок 1.8 – Розумне місто [9]

1.4 Висновки до розділу 1

У даному розділі були розглянуті особливості транспортних мереж та мереж доступу. Варто зазначити, що транспортні мережі забезпечують високошвидкісну передачу даних на великі відстані та включають оптичні мережі (DWDM), Ethernet, MPLS та взаємоз'єднують різні мережі доступу. . Мережі доступу забезпечують підключення кінцевих користувачів до основної мережі. Це включає провідні технології, такі як DSL, FTTx, Ethernet, а також бездротові технології, такі як Wi-Fi, WiMAX. Гібридні мережі, такі як HFC, комбінують переваги різних технологій для забезпечення оптимального зв'язку.

Мережі IoT, передають інформацію з сенсорів та інших пристроїв IoT до мереж доступу.

Розвиток мереж IoT значно змінив сучасну телекомунікаційну інфраструктуру, сприяючи новим підходам до збору, обробки та передачі даних. Мережі IoT інтегруються в існуючу телекомунікаційну інфраструктуру, використовуючи як провідні, так і бездротові технології доступу. Вони сприяють розвитку розумних міст, промислових IoT, медичних систем та багатьох інших сфер, підвищуючи ефективність та точність моніторингу і управління. IoT вимагає забезпечення високого рівня безпеки та надійності для захисту даних і пристроїв, що підключені до мережі.

Архітектура мережі IoT складається з кількох рівнів, кожен з яких виконує специфічні функції для забезпечення надійного збору, обробки та передачі даних.

Розвиток мереж електронних комунікацій та IoT має значний вплив на сучасну телекомунікаційну інфраструктуру, забезпечуючи більш ефективне управління та моніторинг у різних галузях. Інтеграція різноманітних технологій доступу, сенсорних мереж та хмарних сервісів створює надійну та масштабовану систему для передачі та обробки великих обсягів даних, що є ключовим для розвитку Інтернету речей.

2 ПРОТОКОЛИ РІВНЯ ЗАСТОСУНКІВ В МЕРЕЖАХ ІОТ

2.1 Протокол MQTT

MQTT (Message Queuing Telemetry Transport) — є найбільш часто використовуваним протоколом обміну повідомленнями для Інтернету речей. Він розроблений як легкий засіб для обміну повідомленнями у форматі публікації/підписки, який ідеально підходить для підключення віддалених пристроїв з невеликим кодовим простором і мінімальною пропускною здатністю мережі [9;10].

Відправник і одержувач спілкуються за допомогою Тем і відокремлені один від одного. Зв'язком між ними займається брокер MQTT, який фільтрує всі вхідні повідомлення і правильно розподіляє їх серед одержувачів. Сьогодні MQTT використовується в найрізноманітніших галузях, таких як автомобілебудування, виробництво, телекомунікації, нафтогазова промисловість тощо [9;10].

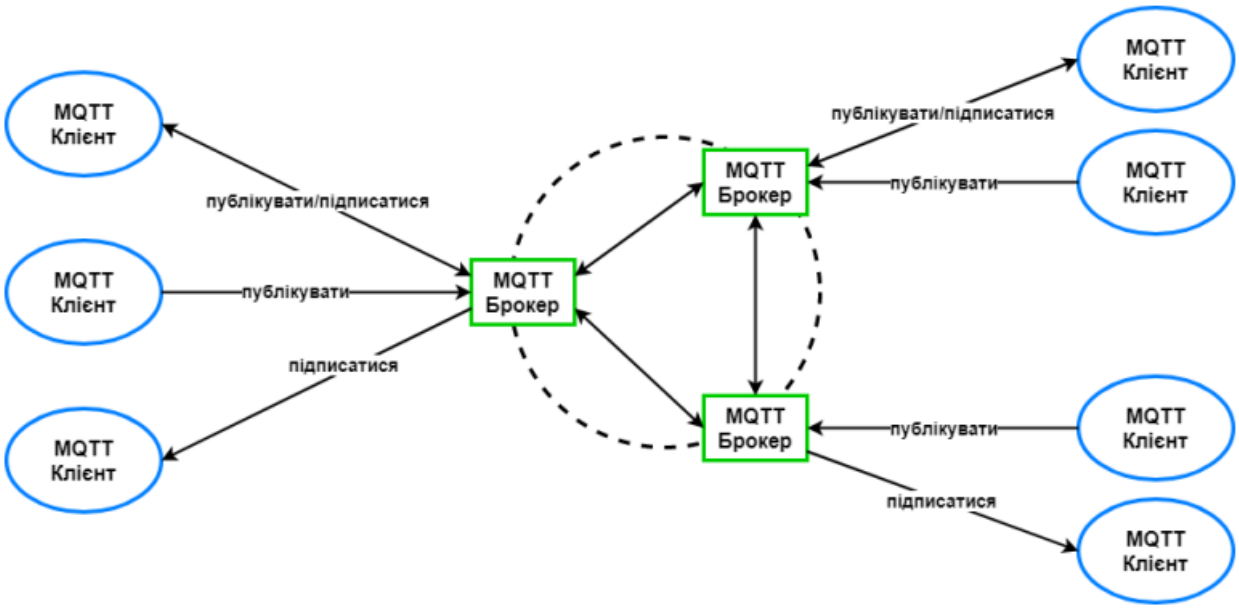

Рисунок 2.1 – Архітектура протоколу MQTT [11]

Модель обміну повідомленнями MQTT базується на темах і підписках. Теми – це рядки, на які публікуються повідомлення та на які користувач підписується. Теми є ієрархічними та можуть містити кілька рівнів, розділених косими рисками, як-от шлях до файлу: *myhome/kitchen/smartdishwasher*

Підписки використовуються для того, щоб вказати, з яких тем клієнт зацікавлений в отриманні повідомлень [10].

Коли клієнт підписується на тему, це, по суті, говорить брокеру, що він зацікавлений в отриманні повідомлень, опублікованих на цю тему. Потім брокер відстежує підписку та пересилає будь-які повідомлення, опубліковані на цю тему, клієнту, на який підписано [10].

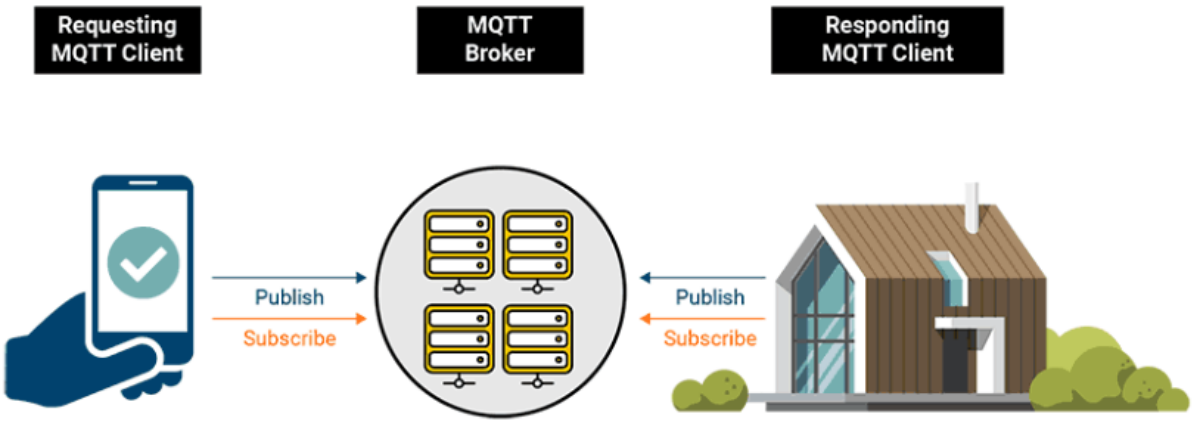

Рисунок 2.2 – Приклад: розумне відкривання дверей за допомогою мобільного пристрою за допомогою MQTT [10]

Важливо зазначити, що клієнт може підписатися на кілька тем одночасно, а тема може мати кілька підписників. Це дозволяє створити гнучку та масштабовану систему обміну повідомленнями [10].

На додаток до тем і підписок, MQTT також підтримує символи підстановки, які можна використовувати для підписки на кілька тем, дотримуючись певного шаблону. Є дватипи символів: однорівневий символ

узагальнення (+), який відповідає одному рівню в темі та багаторівневий символ узагальнення (#), який відповідає всім рівням після вказаного рівня в темі [10].

Модель обміну повідомленнями MQTT забезпечує гнучкий і масштабований спосіб публікації повідомлень і підписки на них, використовуючи теми та підписки. Використання символів узагальнення додає додатковий рівень гнучкості, дозволяючи підписуватися на кілька пов'язаних тем за допомогою однієї підписки [10].

Розуміння моделі обміну повідомленнями MQTT має важливе значення, але не менш важливим – рівень якості обслуговування (QoS – Quality of Service). Даний протокол підтримує три рівні якості обслуговування: QoS 0, QoS 1 і QoS 2 [10].

QoS 0: Найшижчий рівень QoS. Даний рівень забезпечує доставку «максимум один раз», коли повідомлення надсилаються без підтвердження та можуть бути втрачені. Він використовується тоді, коли втрата повідомлення є прийнятною або коли повідомлення не є критичним [10].

Наприклад: QoS 0 може надсилати дані датчиків, якщо випадкова втрата даних суттєво не вплине на загальні результати [10].

QoS 1: Даний рівень забезпечує доставку «принаймні один раз», коли повідомлення підтверджуються та повторно надсилаються за необхідності. За допомогою QoS 1 видавець надсилає повідомлення брокеру, а потім чекає на підтвердження, перш ніж продовжити. Якщо брокер не відповідає протягом встановленого часу, видавець повторно надсилає своє повідомлення. Зазвичай використовується в ситуаціях, коли втрата повідомлень неприпустима, але можна дублювати повідомлення [10].

Наприклад: QoS 1 може підійти для надсилання командних повідомлень на пристрої, де пропущена команда може мати серйозні наслідки, а дубльовані команди — ні [10].

QoS 2: є найвищим рівнем QoS Цей рівень забезпечує доставку «рівно один раз», коли повідомлення підтверджуються та надсилаються повторно, доки вони не будуть отримані абонентом рівно один раз. Використовується в ситуаціях, коли втрата або дублювання повідомлень абсолютно неприпустимі. У QoS 2 видавець і брокер беруть участь у двоетапному процесі підтвердження, де брокер зберігає повідомлення до тих пір, поки воно не буде отримано і підтверджено одержувачем. Даний рівень QoS зазвичай використовується для критично важливих повідомлень, таких як фінансові транзакції або екстрені сповіщення [10].

Важливо зазначити, що вищий рівень QoS зазвичай вимагає більше ресурсів і може призвести до збільшення затримки та мережевого трафіку. Тому важливо вибрати відповідний рівень QoS, виходячи з конкретних потреб користувача програми [10].

MQTT також підтримує збереження повідомлень, що гарантує, цілісність даних у разі збою мережі або сервера. У MQTT стійкість повідомлень досягається шляхом зберігання повідомлень на сервері до тих пір, поки вони не будуть доставлені абоненту [10].

MQTT надає три типи параметрів збереження повідомлень, які представлено нижче:

Непостійний: Це параметр за замовчуванням у MQTT. У цьому режимі повідомлення не зберігаються на сервері і втрачаються, якщо сервер або мережа виходять з ладу. Цей режим підходить для ситуацій, коли повідомлення не є критичними і їх можна легко відновити [10].

Постійна черга: у цьому режимі повідомлення зберігаються на сервері, доки вони не будуть доставлені абоненту. Якщо абонент недоступний, повідомлення стоять у черзі, доки абонент не підключиться знову. Постійність у черзі корисна, коли абонент не завжди підключений до мережі, або якщо абоненту потрібно отримувати всі повідомлення, навіть якщо вони надсилаються, коли абонент не в мережі [10].

Постійний із підтвердженням: Цей режим забезпечує найвищий рівень стійкості повідомлень. У цьому режимі повідомлення зберігаються на сервері до тих пір, поки вони не будуть доставлені абоненту, і абонент повинен підтвердити отримання повідомлення. Якщо абонент не підтверджує отримання, повідомлення надсилається повторно, доки абонент не підтвердить отримання. Цей режим корисний, коли критично важливо забезпечити отримання та обробку повідомлень абонентом [10].

Важливо зазначити, що стійкість повідомлень пов'язана з компромісом з точки зору продуктивності та зберігання. Чим стійкіші повідомлення, тим більше ресурсів для зберігання та обробки потрібно брокеру. Тому важливо вибрати відповідний рівень наполегливості, виходячи з конкретних вимог програми [10].

З точки зору безпеки, MQTT підтримує шифрування TLS для безпечного зв'язку між клієнтами та сервером. Існує кілька стратегій захисту розгортання MQTT, таких як шифрування зв'язку, впровадження надійної автентифікації та контролю доступу тощо [10].

2.2 Протокол DDS

DDS (Data Distribution Service) — це протокол для IoT, який забезпечує мережеву сумісність для підключених машин, корпоративних систем та мобільних пристроїв. Також забезпечує масштабованість, продуктивність і якість обслуговування, що являється необхідним для підтримки додатків IoT. DDS може бути розгорнутий на різних платформах: від малогабаритних пристроїв до хмари, а також підтримує ефективне використання пропускної здатності. Даний протокол забезпечує глобальний простір даних для аналітики та гнучку системну інтеграцію в режимі реального часу [14].

DDS є унікально орієнтованим на дані, що ідеально підходить для промислового Інтернету речей (IIoT). Більшість проміжного програмного

забезпечення працює надсилаючи інформацію між програмами та системами. Орієнтованість на дані гарантує, що всі повідомлення містять чітку інформацію, необхідну додатку для розуміння даних, які він отримує.

Суть орієнтованості на дані полягає в тому, що DDS знає, які дані він зберігає, контролює, а також як ними обмінюватися [12].

Даний протокол бачить локальне сховище даних, яке називається «глобальним простором даних». Для застосунку глобальний простір даних виглядає як локальна пам'ять, доступ до якої здійснюється через API. Користувач пишете в те, що виглядає як локальне сховище. Насправді DDS надсилає повідомлення для оновлення відповідних сховищ на віддалених вузлах [12].

На відміну від інших протоколів публікації/підписки, таких як MQTT, DDS покладається на архітектуру без посередників, а також використовує багатоадресну розсилку, задля забезпечення відмінної якості обслуговування (QoS) і високої надійності своїх програм. Його архітектура публікації/підписки без брокерів добре відповідає обмеженням реального часу для комунікацій IoT та M2M [12].

У домені DDS одиницею обміну інформацією є об'єкти даних у межах Теми. Тема ідентифікується за її назвою, а об'єкт даних — за допомогою атрибутів 'Key). Це схоже на те, як ключові атрибути використовуються для ідентифікації записів у базі даних. Ця цифра концептуальна. DDS обмінюється даними між вузлами і не вимагає посередництва даних сервером або хмарою [12].

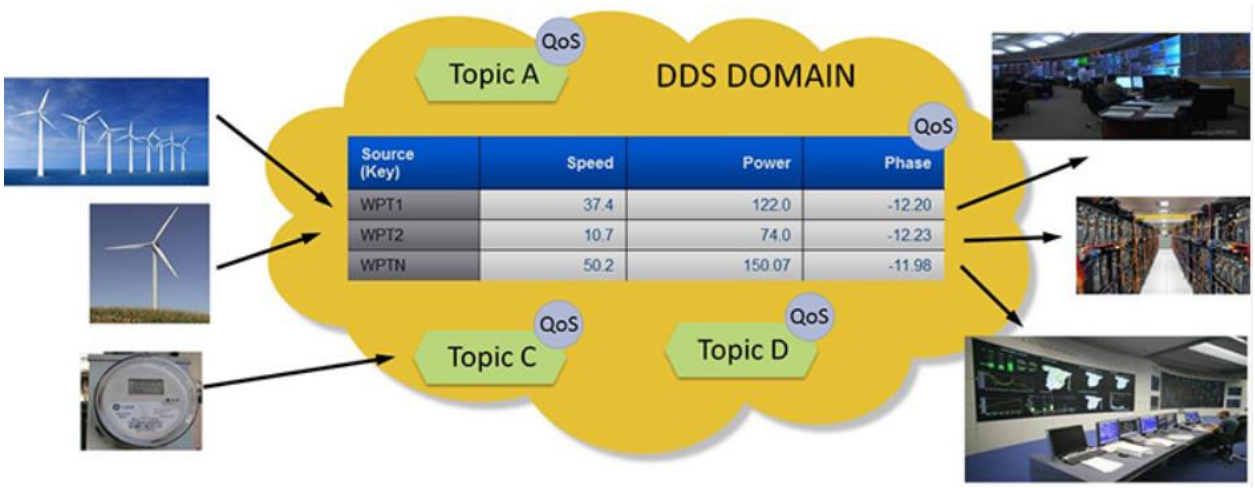

Рисунок 2.3 - DDS [12]

DDS підтримує 23 політики QoS, за допомогою яких розробник може врахувати різноманітні критерії зв'язку, такі як безпека, терміновість, пріоритет, довговічність, надійність тощо [12].

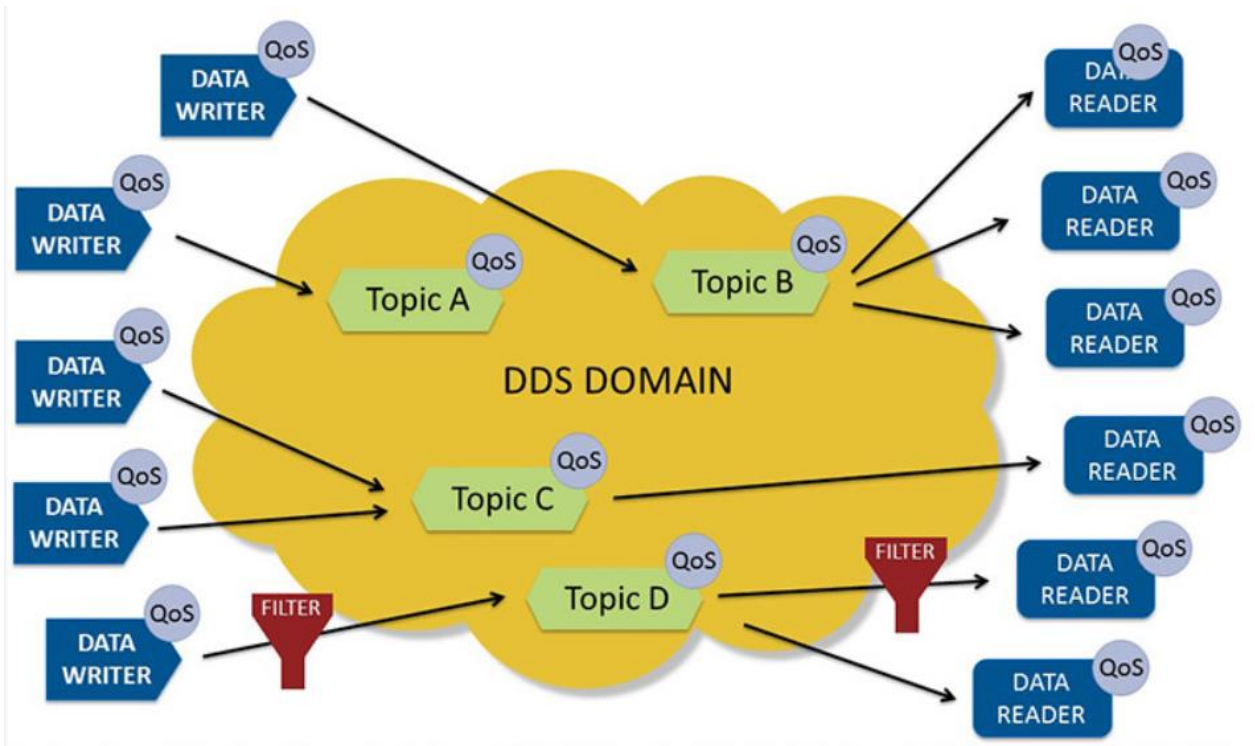

Рисунок 2.4 - 23 політики QoS в протоколі DDS [12]

DDS забезпечує динамічне виявлення видавців і передплатників. Дана процедура робить програми DDS розширюваними. Це означає, що програмі

не потрібно знати або налаштовувати кінцеві точки для зв'язку, оскільки вони автоматично виявляються DDS. Це може бути зроблено під час виконання, а не обов'язково під час проектування або компіляції, що дозволяє справжнє "plug-and-play" для додатків DDS [12].

Це динамічне відкриття йде навіть далі, ніж виявлення кінцевих точок. DDS виявить, чи публікує кінцева точка дані, чи підписується на дані, чи те й інше. Він визначить тип даних, які публікуються або на які користувач підписується. Він також виявить запропоновані видавцем комунікаційні характеристики та запитувані комунікаційні характеристики підписника. Всі ці атрибути враховуються при динамічному виявленні і зіставленні учасників DDS [12].

Учасники DDS можуть знаходитися на одному комп'ютері або в мережі: додаток використовує один і той же DDS API для зв'язку. Оскільки немає необхідності знати або налаштовувати IP-адреси, або враховувати відмінності в архітектурах машин, додавання додаткового учасника зв'язку на будь-якій операційній системі або апаратній платформі стає легким, майже тривіальним завданням [12].

Системи DDS можуть простягатися від Edge до Fog і Cloud. На периферії вони можуть бути використані для високошвидкісного зв'язку між машинами в реальному часі. У системі посередників вони можуть забезпечити надійний, надійний QoS та інформаційні потоки з урахуванням вмісту. Інтеграція цих систем DDS забезпечує масштабований доступ і розподіл інформації на всьому шляху до хмари та всередині неї [12].

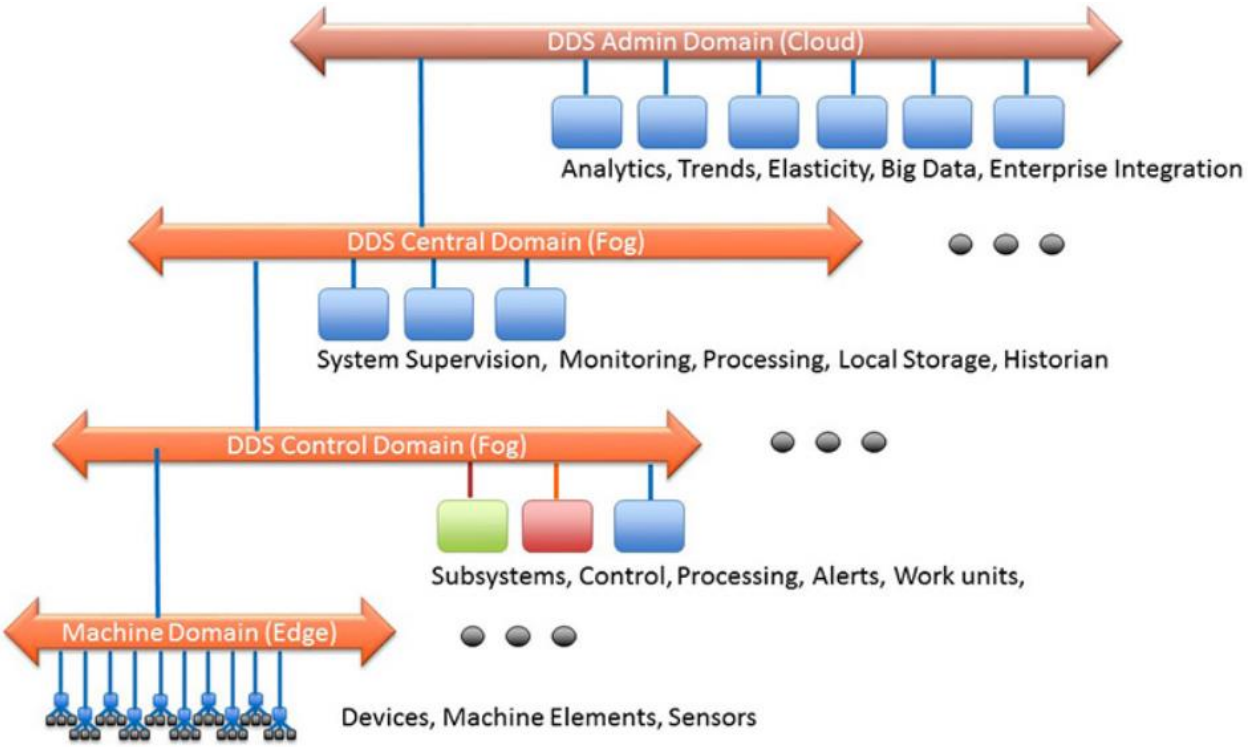

Рисунок 2.5 - Системи DDS та Edge, Fog, Cloud [12]

Захист критично важливих промислових середовищ Інтернету речей вимагає безпеки, яка масштабується від периферії до хмари, між системами та постачальниками. DDS включає в себе механізми безпеки, що забезпечують аутентифікацію, контроль доступу, конфіденційність і цілісність розподілу інформації. DDS Security використовує децентралізовану однорангову архітектуру, яка забезпечує безпеку без шкоди для продуктивності в реальному часі [12;13].

2.3 Протокол CoAP

CoAP (Constrained Application Protocol) - це протокол для взаємодії між пристроями IoT, які мають обмежені ресурси. Дані пристрої мають обмежену пам’ять, сховище і обчислювальну потужністю, а також обмежений заряд батареї та низьку пропускну здатністю [15].

Рисунок 2.6 - План функціонування CoAP [15]

План функціонування CoAP (рисунок 2.6):

1. Клієнт CoAP відправляє запит на сервер CoAP для доступу до ресурсу. Запит включає в себе наступні методи: (GET, POST, PUT, DELETE), а також URI, що ідентифікує ресурс та будь-які додаткові параметри [16].
2. Сервер CoAP отримує запит і обробляє його. Якщо запит дійсний, то сервер генерує відповідь і відправляє її назад клієнтові. Якщо ж запит недійсний, то сервер відправляє відповідь з помилкою [16].
3. Сервер відправляє відповідь назад клієнтові, що містить код стану (наприклад, вміст 2.05, 4.04 Не знайдено), корисне навантаження та будь-які додаткові параметри [16].
4. Клієнт одержує відповідь від сервера і обробляє його. Якщо ж відповідь містить корисне навантаження, то клієнт може використовувати ці дані для виконання бажаної операції [16].

Варто зазначити, що на додаток до базової моделі запит-відповідь, CoAP також включає в себе деякі додаткові функції, що представлені нижче:

1. Клієнт CoAP може спостерігати за ресурсом, відправивши запит спостереження на сервер. Потім сервер буде відправляти повідомлення клієнту кожен раз, коли ресурс змінюється [16].
2. Протокол CoAP підтримує передачу даних по блоках, коли великі ресурси можуть бути розділені на більш дрібні блоки і передані в послідовності запитів і відповідей. Це дозволяє пристроям з обмеженою пам’яттю і пропускною здатністю отримувати доступ до великих ресурсів [16].
3. CoAP підтримує безпеку транспортного рівня дейтаграм (DTLS) задля забезпечення безпечного зв’язку за протоколом UDP. Він також підтримує полегшені механізми безпеки: попередній спільний ключ (PSK) і Необроблений відкритий ключ (RPK) для аутентифікації кінцевих точок і безпечного зв’язку [16].
4. Протокол CoAP надає простий і ефективний механізм для виявлення ресурсів в мережі. Він використовує формат базової посилання для опису ресурсів та їх властивостей [16].

Проблеми безпеки протоколу CoAP та їх усунення:

1. На жаль, CoAP не надає вбудованих механізмів автентифікації, це означає, що будь-хто може надсилати запити на сервер CoAP. Щоб вирішити дану проблему, можна використовувати безпеку транспортного рівня дейтаграм (DTLS) для забезпечення безпечної зв’язку за протоколом UDP. DTLS надає служби аутентифікації і шифрування для захисту зв’язку між клієнтом CoAP і сервером [16].
2. CoAP є уразливим для атак типу “Відмова в обслуговуванні” (DoS), це коли зловмисник завалює сервер запитами, і в результаті чого він стає перевантаженим та перестає відповідати. Щоб запобігти цим атакам, потрібно реалізувати механізми обмеження швидкості та контролю доступу на сервері CoAP. Вони обмежують кількість

запитів, які можуть бути відправлені на сервер та запобігають несанкціонованому доступу [16].

3. Протокол CoAP не надає вбудованих механізмів авторизації, це означає, що будь-хто може надсилати запити на сервер та може отримати доступ до його ресурсів. Вирішення проблеми полягає в тому, що можна реалізувати механізми контролю доступу на сервері. Дані механізми обмежують доступ до певних ресурсів в залежності від особистості клієнта, а також типу надісланого запиту [16].
4. Повідомлення CoAP можуть бути перехоплені і змінені зловмисником, це може призвести до несанкціонованого доступу до ресурсів або даних. Щоб не допустити дану проблему потрібно використовувати механізми забезпечення цілісності повідомлень: код автентифікації повідомлення (MAC) або цифрові підписи. Дані механізми гарантують, що повідомлення не було змінено під час передачі [16].
5. Протокол CoAP надає простий і ефективний механізм виявлення ресурсів у мережі, він може бути використаний зловмисниками для отримання несанкціонованого доступу до ресурсів. Щоб запобігти даній проблемі на виявлення ресурсів потрібно використовувати механізми контролю доступу на сервері CoAP для обмеження доступу до певних ресурсів [16].

2.4 Протокол AMQP

AMQP (Advanced Message Queuing Protocol) — це протокол обміну повідомленнями, призначений для полегшення зв'язку між розподіленими системами. AMQP є відкритим протоколом, що призначений для надійного та безпечного обміну повідомленнями [17].

По суті, AMQP дотримується посередницької схеми обміну повідомленнями (див. рисунок 2.8). У даній схемі брокер повідомлень виступає посередником між видавцем і одержувачем. Спочатку повідомлення публікуються на біржу в брокера. Біржа, діючи як “агент” маршрутизації, пересилає ці повідомлення до відповідної черги, використовуючи свої правила маршрутизації [17].

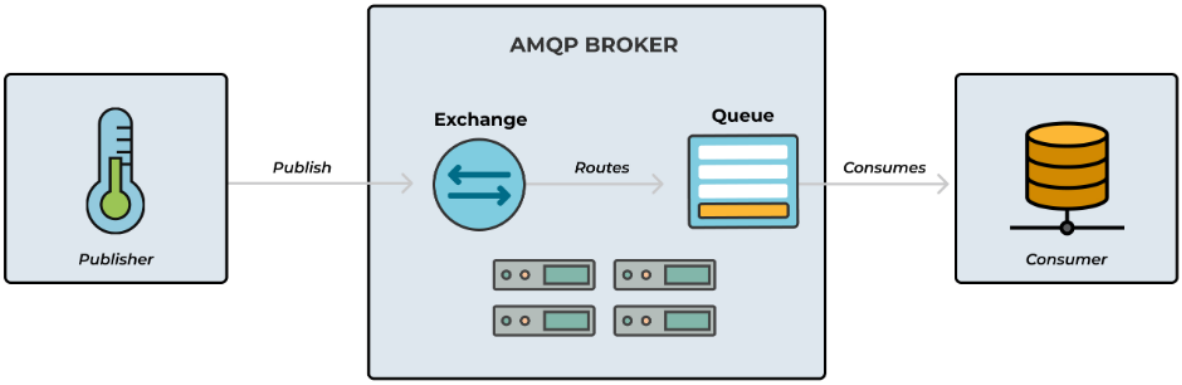

Рисунок 2.8 – Схема роботи протоколу AMQP [17]

AMQP складається із чотирьох важливих компонентів, необхідних для ефективного функціонування. Видавці надсилають повідомлення біржам, які потім їх отримують. Черги – утримують ці повідомлення, до того часу поки вони не будуть готові до використання. Прив'язки – регулюють потік повідомлень між біржами та чергами. Ключі маршрутизації – використовуються видавцями для надсилання повідомлень на біржу [25].

Біржі несуть відповідальність за прийом вхідних повідомлень від видавців, які можуть використовувати різні ключі маршрутизації. Це відбувається задля переконання, що лише певні типи повідомлень входять до певних бірж або черг. Коли видавець надсилає повідомлення біржі, його буде спрямовано відповідно за правилами адміністратора. Крім того, біржа може фільтрувати неактуальну інформацію до того, як вона потрапить у чергу [25].

Черги зберігають усі опубліковані повідомлення до тих пір, поки споживачі їх не прочитають. Тому черга може перебувати в одному з двох станів: активному або неактивному. Активні черги містять дані, що готові до доставки, а неактивні – дані, які ще не були побачені споживачами або обробляються перед входом у систему [25].

Прив'язки контролюють переміщення даних між біржами та чергами, забезпечуючи при цьому безперебійний зв'язок між ними з мінімальними збоями або затримками. Вони надійні, а також можуть бути легко налаштовані в поточних програмах і службах. Адміністратори можуть настроїти прив'язки таким чином, щоб спрямовувати всі повідомлення Exchange до однієї черги або кількох черг, якщо у цьому буде потреба, то для більшої системи. Це можна зробити без додаткового тягаря ручної реалізації або використання інструментів API для розробників [25].

Ключі маршрутизації — це атрибути метаданих, які використовуються для маршрутизації повідомлень у системі обміну повідомленнями. В протоколі AMQP повідомленню присвоюється ключ маршрутизації, коли воно надсилається біржі. Даний ключ маршрутизації потім застосовується біржею для визначення черг, куди має бути надіслано повідомлення. Прив'язуючи черги до обмінів певними ключами маршрутизації, повідомлення можна вибірково маршрутизувати на основі вмісту ключа маршрутизації. Цей функціонал дозволяє ефективно та адаптивно розповсюджувати повідомлення в системах, що працюють на базі AMQP [25].

При комбінованому використанні дані елементи створюють зручну платформу, яка є доступною для великомасштабних і складних систем. Це робить протокол AMQP найкращим вибором для забезпечення надійного зв'язку через мережі [25].

AMQP підтримує схему обміну повідомленнями «публікація-підписка». Повідомлення від одного видавця транслюється на кілька абонентів [17].

Також AMQP підтримує асинхронний зв'язок. Видавець надсилає повідомлення брокеру не чекаючи відповіді від нього. У результаті і видавцеві, і отримувачу не доведеться перебувайте в Інтернеті одночасно, щоб мати можливість спілкуватися між собою [17].

Протокол AMQP має декілька рівнів QoS для забезпечення надійності доставки повідомлень. Зберігання повідомлень на диску та повторна передача повідомлень у випадку збоїв [17].

AMQP підтримує аутентифікацію та шифрування за допомогою SSL/TLS для захисту передачі даних між клієнтами та серверами [17].

AMQP є універсальним і найкраще підходить для сценаріїв, які вимагають надійного, безпечного та ефективного зв'язку між різними системами і пристроями. Іншими словами, протокол чудово підходить для використання повідомлень, що мають бути доставленим без втрат і дублювання [17].

Багатий набір функцій AMQP та складна маршрутизація його механізму забезпечує розширений функціонал за рахунок збільшенні накладних витрат. AMQP вимагає більш об'ємної мережі площі та обчислювальних ресурсів [17].

Приклад: у світі фінансів своєчасний і точний обмін повідомленнями необхідні для проведення транзакцій та управлінням даними, тож AMQP гарантує, що повідомлення будуть надіслані адресатам без будь-яких втрат [17].

2.5 Протокол HTTP

Протокол передачі гіпертексту (HTTP) є основою Всесвітньої павутини його використовують для завантаження веб-сторінок за допомогою гіпертекстових посилань [18].

HTTP (Hypertext Transfer Protocol) – це протокол прикладного рівня, призначений для передачі інформації між мережевими пристроями. Типовий потік через HTTP передбачає, що клієнтська машина робить запит до сервера, який потім надсилає повідомлення у відповідь [18].

Він є основою будь-якого обміну даними в Інтернеті, тож являється клієнт-серверним протоколом, це означає, що запити ініціюються одержувачем, як правило – веб-браузером. Повний документ відтворюється з різних отриманих вкладених документів: тексту, зображень, відео, сценаріїв тощо (див. рисунок 2.9) [19].

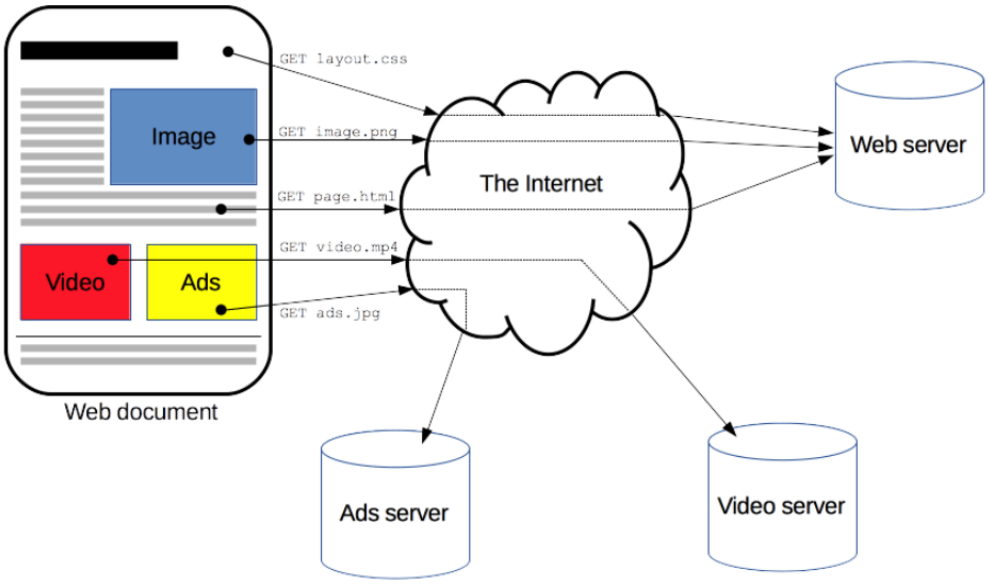

Рисунок 2.9 – HTTP – клієнт-серверний протоколо [19]

Клієнти та сервери спілкуються, обмінюючись окремими повідомленнями (на відміну від потоку даних). Повідомлення, які

надісилаються клієнтом, як правило веб-браузером, називаються запитами, а повідомлення – сервером як відповідь, називаються відповідями [19].

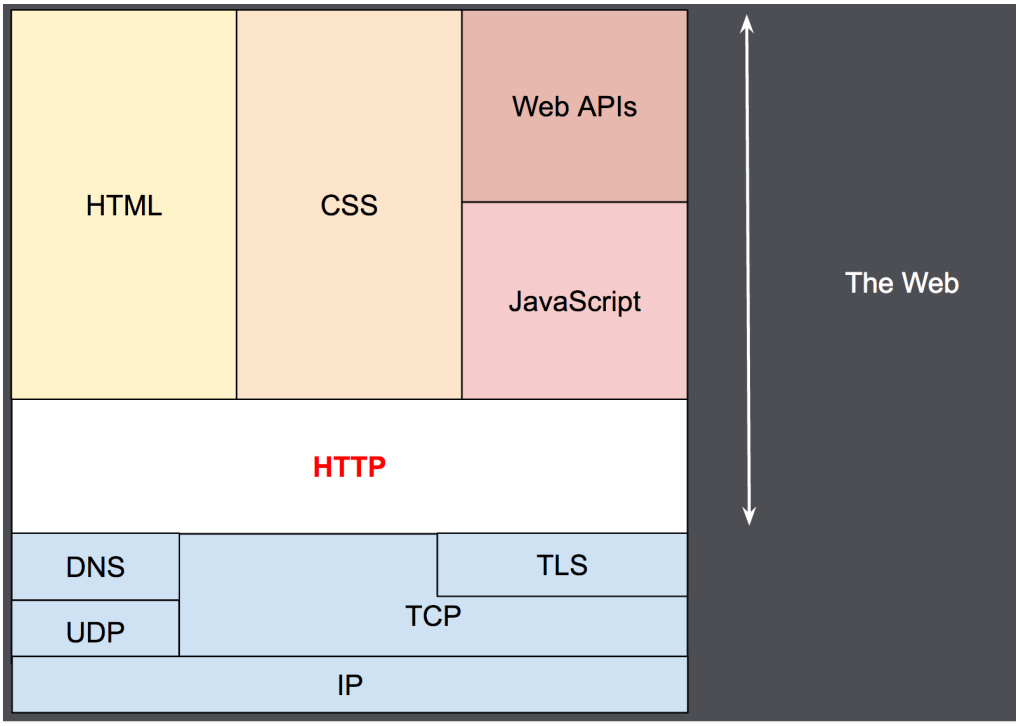

Рисунок 2.10 – Протокол HTTP [19]

Розроблений на початку 1990-х років, HTTP є розширюваним протоколом, який розвивався з часом. Він надсилається через TCP (див. рисунок 2.10) або через TCP-з'єднання, зашифроване TLS, хоча можна використовувати будь-який інший надійний транспортний протокол. Завдяки своїй розширюваності, HTTP використовується не тільки для отримання гіпертекстових документів, але також зображень та відео або для публікації контенту на серверах, як у випадку з результатами HTML-форм. HTTP також можна використовувати для вибірки частин документів для оновлення веб-сторінок [19].

Методи HTTP

- GET використовується для отримання даних із вказаного ресурсу. Він не повинен мати побічних ефектів і використовується для отримання веб-сторінок, зображень [20].
- POST використовується для надсилання даних для обробки вказаним ресурсом. Він підходить для подання форм, завантаження файлів і створення нових ресурсів [20].
- PUT використовується для оновлення або створення ресурсу на сервері. Він замінює весь ресурс даними, наданими в тілі запиту [20].
- PATCH схожий на PUT, але використовується для часткових модифікацій ресурсу: оновлює певні поля ресурсу, а не замінює весь ресурс [20].
- DELETE використовується для видалення ресурсів з сервера [20].
- HEAD подібний до GET, але отримує лише заголовки відповідей, корисний для перевірки властивостей ресурсу без передачі повного вмісту [20].
- OPTIONS використовується для отримання параметрів зв'язку, доступних для ресурсу, включаючи підтримувані методи та заголовки [20].
- TRACE: Використовується з метою налагодження для повторення отриманого запиту клієнту, хоча використовується рідко з міркувань безпеки [20].
- CONNECT: використовується для встановлення тунелю до сервера через проксі-сервер HTTP, зазвичай використовується для з'єднань SSL/TLS [20].

HTTP – це протокол запит-відповідь, це означає, що на кожен запит, надісланий клієнтом (зазвичай веб-браузером), сервер відповідає відповідною відповіддю. Основний потік циклу HTTP-запит-відповідь виглядає наступним чином (див. рисунок 2.11):

1. Клієнт (зазвичай веб-браузер) ініціює процес, надсилаючи HTTP-запит на сервер. Цей запит включає метод запиту (GET, POST, PUT тощо), цільовий URI (уніфікований ідентифікатор ресурсу, наприклад, URL), заголовки та необов'язкове тіло запиту [20].
2. Сервер отримує запит і обробляє його на основі запитуваного методу та ресурсу. Дана процедура може включати отримання даних із бази даних, виконання сценаріїв на стороні сервера або виконання інших операцій [20].
3. Після обробки запиту сервер надсилає HTTP-відповідь назад клієнту. Відповідь містить код стану (наприклад, 200 OK, 404 Not Found), заголовки відповідей і необов'язкове тіло відповіді, яке містить запитувані дані або вміст [20].
4. Клієнт отримує відповідь сервера та обробляє її відповідним чином. Наприклад, коли відповідь містить HTML-сторінку, браузер відобразить її. Якщо це зображення або інший мультимедійний файл, браузер відобразить або обробить його належним чином [20].
5.

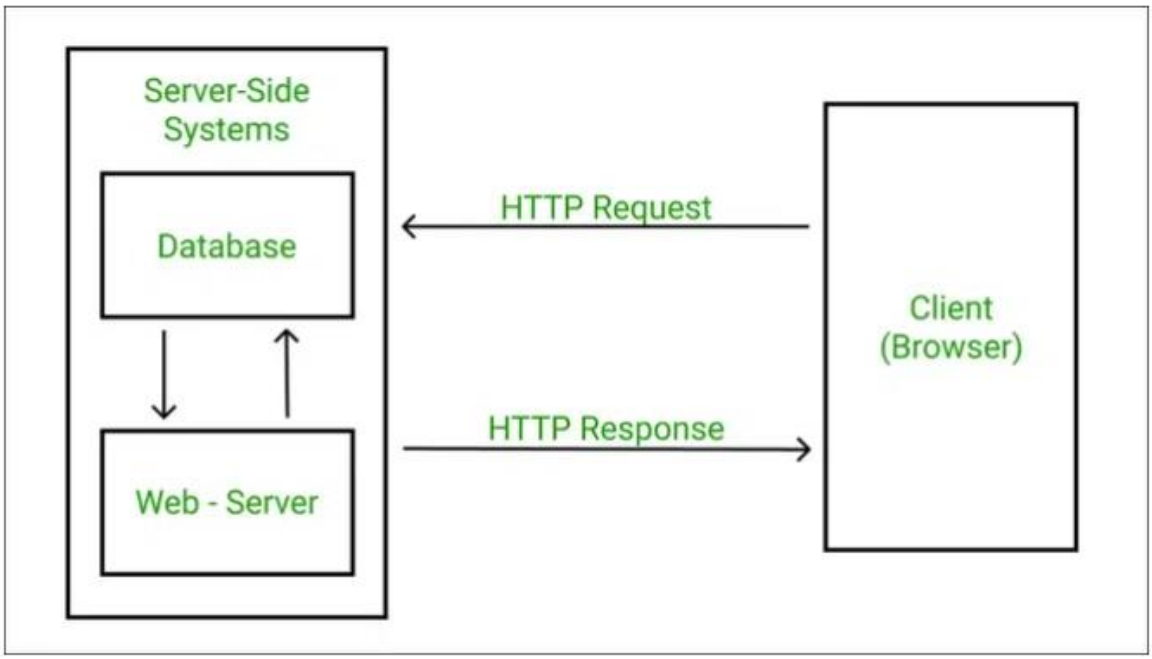

Рисунок 2.11 – HTTP запит/відповідь [20]

В оригінальній специфікації HTTP-запити створюють і закривають TCP-з'єднання. У новіших версіях протоколу HTTP, постійне з'єднання дозволяє кільком запитам HTTP передавати постійне TCP-з'єднання, при цьому покращуючи споживання ресурсів. У контексті DoS- або DDoS-атак HTTP-запити у великих кількостях можуть використовуватися для здійснення атаки на цільовий пристрій і вважаються частиною атак прикладного рівня [18].

2.6 Аналіз та порівняння протоколів

Проаналізувавши всі протоколи в попередніх розділах можна зробити наступну таблицю з їх порівнянням.

Таблиця 2.1 – Порівняння протоколів

	MQTT	CoAP	DDS	AMQP	HTTP
Клієнт-серверна архітектура	так	так	так	так	так
Механізм публікації/підписки	так	так	так	так	ні
Запит/відповідь	ні	так	ні	так	так
Транспортний протокол	TCP	UDP	TCP/ UDP	TCP	TCP
Безпека	SSL	DTLS	SSL/DTLS	TLS	SSL
QoS	так	так	так	так	ні
Розмір заголовку (байт)	2	4	-	8	-

2.7 Висновки до розділу 2

Отже, протоколи рівня застосунків в мережах IoT відіграють ключову роль у забезпеченні ефективного та надійного обміну даними між пристроями та системами. Вибір протоколу залежить від конкретних вимог додатка, таких як продуктивність, надійність, ефективність використання ресурсів та сумісність з існуючими інфраструктурами. Розуміння

особливостей та переваг кожного протоколу дозволяє оптимально інтегрувати їх у різні IoT-рішення, забезпечуючи ефективну взаємодію та обмін даними в сучасних телекомунікаційних мережах.

MQTT є легким протоколом обміну повідомленнями, який використовує модель публікації/підписки. Протокол розроблений для пристроїв з обмеженими ресурсами та мереж з низькою пропускною здатністю, що робить його ідеальним для IoT. Основні переваги включають низьку затримку, мінімальне використання смуги пропускання та високу надійність доставки повідомлень.

DDS є стандартом для обміну даними в реальному часі з високою продуктивністю та надійністю. Використовує модель публікації/підписки та підтримує широкий спектр якості обслуговування (QoS) параметрів. Протокол підходить для критично важливих додатків, де потрібна низька затримка та висока масштабованість, таких як промисловий IoT та автоматизація.

CoAP розроблений для роботи в обмежених умовах, таких як низька пропускна здатність мережі та обмежені ресурси пристроїв. Протокол використовує модель клієнт/сервер. CoAP забезпечує ефективну передачу даних та низьке енергоспоживання, що робить його підходящим для сенсорних мереж та систем домашньої автоматизації.

AMQP забезпечує надійний обмін повідомленнями між різними додатками та системами. Протокол підтримує різні моделі маршрутизації та забезпечує високу надійність доставки повідомлень. AMQP підходить для використання в корпоративних додатках, де важлива інтеграція з існуючими системами та забезпечення надійності.

HTTP є основним протоколом для обміну даними в Інтернеті та використовується для взаємодії веб-додатків. Протокол використовує модель клієнт/сервер та забезпечує підтримку широкого спектра додатків через простоту та інтероперабельність. HTTP є зручним для використання в IoT-

додатках, де важлива інтеграція з веб-сервісами та існуючими інфраструктурами.

3 ВИКОРИСТАННЯ ПРОТОКОЛІВ РІВНЯ ЗАСТОСУНКІВ

3.1 Приклади використання протоколів рівня застосунків у різних галузях

Протокол MQTT широко використовується в додатках Інтернету речей, промислового Інтернету речей (IIoT) та M2M. Нижче представлено 5 прикладів.

MQTT використовується для підключення пристроїв у розумному будинку: включаючи лампочки, камери спостереження та інші прилади. Це дозволяє користувачам дистанційно досить зручно керувати своїми домашніми пристроями за допомогою мобільного додатку [10].

Рисунок 3.1 – Приклад: включення лампочки за допомогою Google Assistant [21]

Даний протокол використовується для підключення машин і датчиків на заводах та інших промислових об'єктах. Дана процедура дозволяє здійснювати моніторинг і контроль процесів у режимі реального часу, що може підвищити ефективність і скоротити час простою [10].

MQTT також використовується в сільському господарстві, а саме в точному землеробстві для моніторингу рівня вологості ґрунту, погодних умов і росту сільськогосподарських культур. Дані технології допомагають фермерам оптимізувати процеси зрошення та інші методи управління посівами [10].

MQTT використовується у сфері медицини: для підключення медичних пристроїв і датчиків. Наприклад глюкометри та пульсометри, дані пристрої допомагають медичному персоналу ефективніше надавати допомогу хворим, здійснюючи віддалений моніторинг їхнього стану здоров’я [10].

І п’ята сфера використання даного протоколу – це транспорт. MQTT використовується в підключених автомобілях та інших транспортних системах, задля забезпечення відстеження та моніторингу транспортних засобів у режимі реального часу. Дане застосування протоколу підвищує безпеку та допомагає оптимізувати транспортний потік [10].

Протокол DDS використовується в наступних галузях:

DDS надає унікальну можливість задовольняти вимоги до розподілу даних у режимі реального часу великомасштабних, складних систем управління транспортом. Це дозволяє зручно в режимі реального часу координувати дані телеметрії з іншими даними датчиків для оптимізації складних операцій залізничного, вантажного транспорту та автопарку. Завдяки багатій інформації DDS дозволяє користувачам доставляти більше товарів вчасно за нижчою ціною, покращуючи якість обслуговування та знижуючи витрати на ланцюжок поставок [22].

Дану галузь можна доповнити декількома прикладами використання даного протоколу:

Автономні комунікації транспортних засобів

Aptiv Autonomous Mobility використовує DDS для забезпечення безпечного зв'язку в автомобілі для своїх автономних транспортних засобів.

Connext DDS – це структура, заснована на стандартах, яка підтримує розробку автономних транспортних засобів від досліджень до виробництва, пропонуючи компанії першого рівня рішення з найнижчим рівнем ризику. Aptiv використовує Connext DDS у своїх складних автономних системах як безпечне проміжне програмне забезпечення для обміну та зберігання даних, забезпечуючи при цьому відкриту платформу для більш плавної інтеграції програмних модулів [22].

Розумні автомобілі Volkswagen

Система допомоги водієві та інтегрована безпека Volkswagen використовує DDS для поєднання радарів, лазерних далекомірів і відео для сприяння безпечній експлуатації. Він відстежує очі водія, щоб виявити сонливість, а також виявляє виїзд зі смуги руху, уникає зіткнень і допомагає утримувати автомобіль у своїй смузі [22].

Рисунок 3.2 – Приклад: відстеження очей водія [22]

Управління автопарком

Wi-Tronix використовує DDS для бездротового моніторингу цінних мобільних активів, таких як вантажівки з причепами, локомотиви, промислове обладнання та морські судна [22].

DDS використовують у сфері розумної енергетики, адже даний протокол забезпечує високоякісну, масштабовану інформаційну інфраструктуру з низькою затримкою в режимі реального часу. Це допомагає підвищити продуктивність, якість даних і заощадити кошти [22].

Дану галузь можна доповнити декількома прикладами використання даного протоколу:

Великомасштабна система термоядерного реактора

DDS надає системі дистанційного керування Центру термоядерної енергетики Калхема (CCFE) масштабованість, якість обслуговування та інтеграцію, необхідні для підтримки системи диспетчерського управління та збору даних, яка ремонтує та налаштовує компоненти реактора та прилади в дослідницькому термоядерному реакторі [22].

Використання DDS в інтелектуальних енергетичних системах наступного покоління

Платформа розподіленого інтелекту Duke Energy з DDS забезпечує точний контроль і зменшує переривчастість розподілених енергетичних ресурсів, надає можливість незалежного масштабування за потреби без необхідності розгортання в масштабах всієї системи, знижує витрати бізнесу за рахунок скорочення часу та зусиль на інтеграцію [22].

Розподілена генерація електроенергії, ВІТРОЕНЕРГЕТИКА SIEMENS

DDS забезпечує швидке керування у вітрових турбінах і розподілене пом'якшення поривів по всьому масиву [22].

Протокол DDS успішно використовують в медицині. Адже він дозволяє безпечно обмінюється даними про пацієнтів, також використовується у створенні інтелектуальних клінічних інформаційних систем. Даний протокол підключає пристрої в лікарняних палатах, інтегрує системи всієї лікарні, а також підключається до хмарних і мобільних технологій. Поширення інтелектуальних медичних пристроїв створює нові проблеми інтеграції для мереж пристроїв. DDS надає перевірене рішення для безпечної передачі

даних у режимі реального часу між медичними пристроями та іншими інформаційними системами [22].

DDS забезпечує роботу МРТ і КТ

Системи обробки зображень вимагають величезних потоків даних і високої продуктивності. DDS контролює та оптимізує використання мережі, щоб обробляти мегабайти навантаження без втрати даних [22].

Протокол DDS використовується в автоматизації технологій та виконує наступні функції: забезпечує інтеграцію, масштабованість і гнучкість для підтримки складних, високотехнологічних і розподілених систем управління в реальному часі. Архітектури, засновані на Інтернет-протоколі, забезпечують модульність додатків диспетчерського управління та збору даних (SCADA) і роблять системи більш гнучкими та простими в оновленні [22].

Загальна платформа для автоматизації майнінгових систем

DDS використовується як основа для обміну даними загальної програмної платформи, яку Atlas Copco розгорне в усіх продуктових лініях гірничодобувної промисловості. DDS забезпечує Atlas Copco виняткову продуктивність, масштабованість, надійність та доступність операційної системи, необхідні для їхньої платформи [22].

Інтеграція та керування машинами

DDS забезпечує швидку доступність даних, допомагаючи керувати системою Komatsu 14CM Continuous Miner, і надсилає дані вгору по шахті для моніторингу поверхні в режимі реального часу [22].

Рисунок 3.3 – Приклад: система Komatsu 14CM Continuous Miner [22]

Також DDS використовується і в розумних містах, де відповідає багатогранним вимогам управління міським простором у режимі реального часу. Даний протокл забезпечує повсюдний доступ до даних у режимі реального часу з різних датчиків, що дозволяє відстежувати, контролювати та аналізувати кожен аспект міста. Широка доступність інформації допомагає стимулювати інновації в обслуговуванні; покращення якості життя мешканців та створення продуктивного середовища для бізнесу [22].

Загальноміська інформаційна основа, що підтримує спектр нових електронних послуг для міста

DDS використовується як інформаційна основа в режимі реального часу, надаючи місту Ніцца нові інтелекти для координації та підключення всіх служб паркування та мобільності, а також покращення якості навколишнього середовища в місті [22].

Рисунок 3.4 – Приклад: місто Ніцца з'єднаний бульвар [22]

Протокол CoAP використовується в наступних галузях:

CoAP застосовується в системах автоматизації розумного будинку завдяки низьким накладним витратам і високій надійності. У цих системах різні пристрої, такі як освітлення, термостати та камери відеоспостереження, можуть обмінюватися даними за допомогою протоколу CoAP [23].

Рисунок 3.5 - Системи автоматизації розумного будинку [24]

У промислових додатках IoT надійність та ефективність мають вирішальне значення. Такі пристрої, як датчики та виконавчі механізми, можуть обмінюватися даними за допомогою CoAP, що дозволяє в режимі реального часу відстежувати та контролювати промислові процеси. Підтримка протоколом багатоадресного зв'язку особливо корисна в цих сценаріях, оскільки дозволяє одному пристрою спілкуватися з кількома іншими одночасно [23].

Протокол CoAP все більш застосовується у носимих пристроях і додатках для охорони здоров'я. Ці програми часто включають невеликі пристрої, що живляться від батарейок, які повинні взаємодіяти один з одним або з центральним сервером. Низькі вимоги CoAP до накладних витрат і потужності роблять його корисним для таких типів застосувань [23].

CoAP використовується в системах енергетичного менеджменту, де він дозволяє здійснювати моніторинг і контроль використання енергії в режимі реального часу. Такі пристрої, як інтелектуальні лічильники та контролери управління енергією, можуть використовувати протокол для зв'язку один з одним і з центральним сервером, забезпечуючи високий рівень контролю над споживанням енергії [23].

Протокол AMQP використовується в різноманітних реальних додатках, що охоплюють як великі, так і малі масштаби. Популярні програми обміну повідомленнями, такі як WhatsApp, Slack і Skype, використовують AMQP, щоб гарантувати надійну доставку та отримання повідомлень адресатам. Розподілені системи, такі як Apache Kafka, використовують протокол для встановлення безпечного зв'язку між кількома системами через мережі, мінімізуючи помилки та простої. Системи черги повідомлень з відкритим вихідним кодом, такі як RabbitMQ, покладаються на AMQP за свої надійні можливості обміну повідомленнями. Відомі хмарні провайдери, такі як Amazon Web Services (AWS), також використовують AMQP для полегшення зв'язку між різними службами у своїй хмарній інфраструктурі [25].

AMQP також стає все більш поширеним у створенні додатків Інтернету речей (IoT). Оскільки пристрої IoT часто вимагають зв'язку через нестабільні бездротові мережі, наявність надійних протоколів обміну повідомленнями має вирішальне значення. Використання AMQP дозволяє пристроям IoT безпечно та бездоганно обмінюватися даними без будь-яких перебоїв [25].

Протокол HTTP широко використовується в IoT для передачі даних між пристроями і серверами.

Приклади використання HTTP:

Смарт-термостати, такі як Nest, використовують HTTP для комунікації з хмарними серверами. Вони відправляють дані про температуру, вологість та інші параметри для аналізу і управління кліматом у приміщенні.

Рисунок 3.6 - Смарт-термостат Nest [26]

Системи смарт-освітлення, як-от Philips Hue, використовують HTTP для управління освітленням через веб-інтерфейс або мобільний додаток. Користувачі можуть вмикати, вимикати або змінювати яскравість світла з будь-якої точки світу.

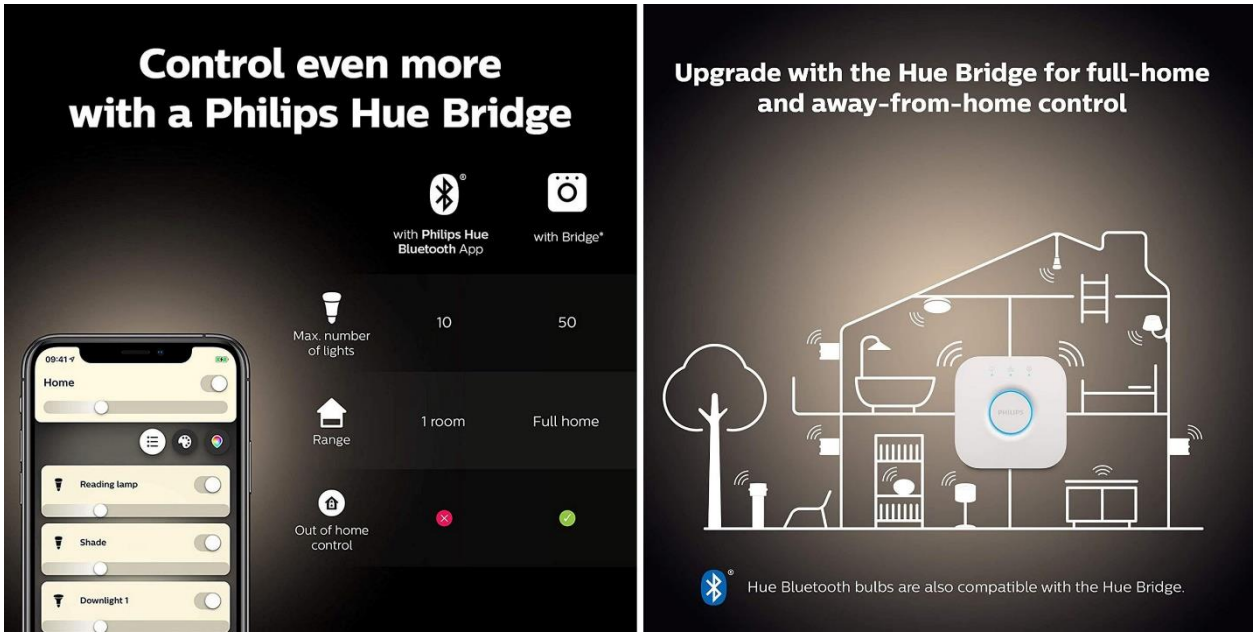

Рисунок 3.7 – Інтерфейс системи Hue [27]

Рисунок 3.8 - Смарт-освітлення Philips Hue [27]

Смарт-камери та системи безпеки, такі як Ring або Arlo, використовують HTTP для передавання відеопотоків і зображень на сервери, де вони зберігаються і доступні для перегляду користувачами через додаток або веб-інтерфейс.

Рисунок 3.9 - Смарт-камера Arlo [28]

Смарт-замки, такі як August Smart Lock, використовують HTTP для управління доступом до приміщень. Користувачі можуть віддалено відмикати або замикати двері через мобільний додаток або веб-інтерфейс.

Рисунок 3.10 – Смарт-замок August Smart Lock [29]

Пристрої для моніторингу здоров'я, такі як фітнес-трекери та смарт-годинники, використовують HTTP для передавання даних про фізичну

активність, частоту серцевих скорочень, сон та інші параметри на сервери для подальшого аналізу та зберігання.

Розумні побутові прилади, такі як пральні машини, холодильники та кавоварки, використовують HTTP для комунікації з хмарними серверами. Користувачі можуть віддалено керувати цими приладами через додаток або веб-інтерфейс.

Рисунок 3.11 – Розумне освітлення [30]

Системи розумного міського освітлення використовують HTTP для управління вуличними ліхтарями. Дані про рівень освітленості, рух транспорту та пішоходів передаються на сервер для оптимального управління освітленням.

Рисунок 3.12 – Розумне вуличне освітлення [31]

Системи точного землеробства використовують HTTP для передавання даних з датчиків вологості ґрунту, температури та інших параметрів на сервери для аналізу та прийняття рішень щодо зрошення та удобрення.

Смарт-датчики для моніторингу якості повітря, води та інших параметрів навколишнього середовища використовують HTTP для передавання зібраних даних на сервери, де вони аналізуються і зберігаються.

Розумні лічильники для газу, води та електроенергії використовують HTTP для передачі даних споживання на сервери постачальників послуг, що дозволяє проводити точний облік та управління ресурсами.

Рисунок 3.13 – Розумний лічильник електроенергії з Wi-Fi D103-600 [32]

3.2 Хмарна обробка

IoT — це мережа підключених пристроїв, які збирають і обмінюються даними, тоді як хмарні обчислення надають обчислювальні ресурси та послуги на вимогу через Інтернет [33].

Інтернет речей і хмарні обчислення доповнюють один одного, причому хмарні обчислення служать центральним вузлом для зберігання та управління даними в системах IoT. Дана інтеграція забезпечує ефективне зберігання даних, спрощений аналіз за допомогою інструментів на основі штучного інтелекту та зниження витрат на операції. Додатки IoT та хмарних обчислень охоплюють різні галузі, включаючи розумні міста, охорону здоров'я та виробництво, підвищуючи ефективність та продуктивність у різних секторах [34] (Рис. 3.14).

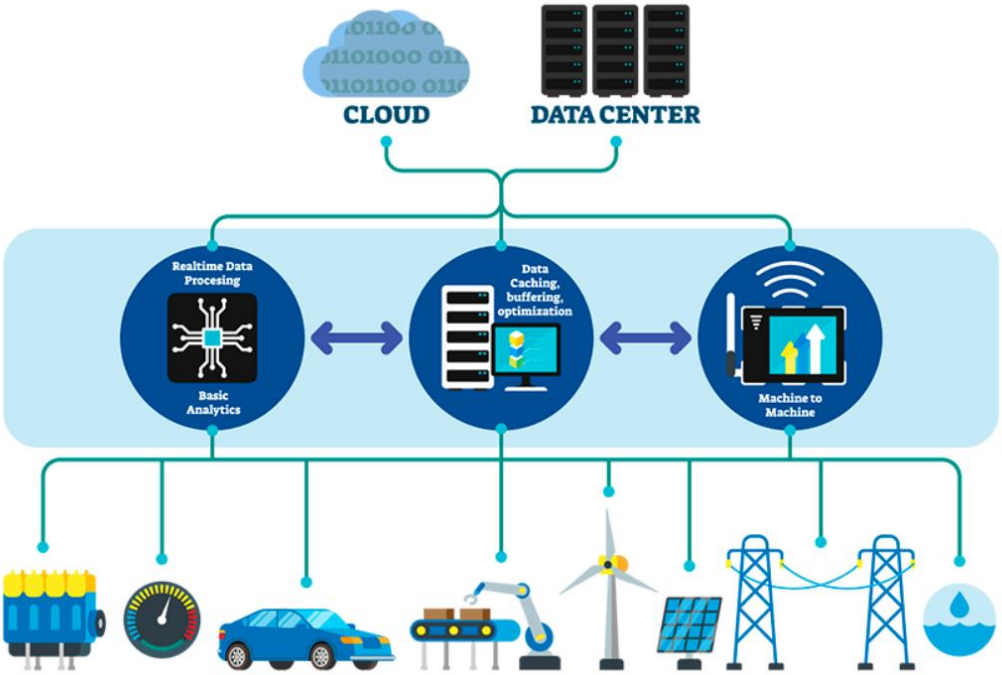

Рисунок 3.14 – IoT і хмарні технології [34]

Величезний обсяг даних, що генеруються щодня пристроями IoT, такими як розумні термостати, фітнес-браслети, можна зберігати та керувати ними в хмарі. Це дозволяє використовувати переваги масштабованості, гнучкості та низької вартості хмари. За допомогою аналітичних інструментів на основі штучного інтелекту можна отримати важливу інформацію з цих необроблених даних, і все це в хмарі [33].

Наприклад: пристрої IoT, які надсилають дані у хмару, можуть відстежувати та вивчати моделі дорожнього руху в місті. Це допомагає містобудівникам знаходити способи зменшити затори на дорогах та покращити життя міста [33].

3.3 Взаємодія з протоколами інших рівнів

Взаємодія MQTT:

1. Для передачі даних через мережу MQTT використовує транспортні та мережеві протоколи, такі як TCP/IP.

TCP (Transport Control Protocol): MQTT використовує TCP для забезпечення надійної передачі повідомлень між клієнтами і брокером. TCP гарантує, що дані доставляються в правильному порядку і без втрат.

IP (Internet Protocol): IP відповідає за адресацію і маршрутизацію пакетів даних у мережі. В MQTT IP використовується для доставки TCP-пакетів між пристроями [35].

2. На канальному рівні використовуються протоколи, що забезпечують передачу даних через фізичні канали зв'язку.

Ethernet/Wi-Fi: MQTT може працювати через Ethernet або Wi-Fi для підключення до локальних мереж.

Bluetooth Low Energy (BLE):У деяких випадках MQTT може бути використано через BLE для забезпечення передачі даних на короткі відстані [35].

3. Фізичний рівень визначає апаратні аспекти передачі даних, такі як типи кабелів, радіосигнали тощо.

802.11 (Wi-Fi): Використовується для бездротового зв'язку між пристроями, що підтримують MQTT.

802.3 (Ethernet): Використовується для проводового зв'язку в локальних мережах [35].

Взаємодія DDS:

1. DDS може використовувати різні транспортні протоколи для передачі даних між додатками:

TCP: DDS може використовувати TCP для надійної передачі даних між додатками на різних пристроях у мережі.

UDP: DDS також може використовувати UDP для швидкої передачі даних у реальному часі, особливо в сценаріях, де важлива низька затримка [36].

2. DDS взаємодіє з протоколами мережевого рівня, такими як IP , для маршрутизації і передачі пакетів даних між пристроями у мережі.

IP: DDS використовує IP для визначення адреси джерела та призначення даних, що передаються через мережу [36].

3. DDS не прямо взаємодіє з фізичними протоколами, такими як Ethernet або Wi-Fi. Він використовує транспортний рівень для забезпечення передачі даних через різні фізичні канали зв'язку [36].

Взаємодія CoAP:

1. CoAP використовує різні транспортні протоколи для передачі даних:

UDP : CoAP зазвичай працює поверх UDP для передачі даних. UDP забезпечує швидку передачу даних без необхідності надійності доставки, що особливо важливо для обмежених пристроїв.

DTLS (Datagram Transport Layer Security): Для забезпечення безпеки передачі даних, CoAP може використовувати DTLS поверх UDP, щоб забезпечити зашифровану та аутентифіковану комунікацію [36].

2. CoAP взаємодіє з протоколами мережевого рівня для адресації та маршрутизації пакетів даних:

IPv4 або IPv6 : CoAP використовує IP для визначення джерела та призначення даних, що передаються через мережу [36].

3. CoAP не прямо взаємодіє з фізичними протоколами, такими як Ethernet або Wi-Fi. Він використовує транспортний рівень для передачі даних через різні фізичні канали зв'язку [36].

Взаємодія AMQP:

1. AMQP використовує транспортні протоколи для передачі повідомлень:

TCP: AMQP використовує TCP для надійної і забезпеченої передачі повідомлень між клієнтами та серверами [37].

2. AMQP взаємодіє з протоколами мережевого рівня для передачі даних через мережу:

IP: AMQP використовує IP для визначення джерела та призначення даних, що передаються через мережу [37].

3. AMQP не прямо взаємодіє з фізичними протоколами, такими як Ethernet або Wi-Fi. Він використовує транспортний рівень для передачі даних через фізичні канали зв'язку [37].

Взаємодія HTTP:

1. HTTP використовує транспортні протоколи для передачі даних:

TCP: HTTP використовує TCP для надійної та забезпеченої передачі даних між клієнтами та серверами.

2. HTTP взаємодіє з протоколами мережевого рівня для маршрутизації та передачі даних через мережу:

IP: HTTP використовує IP для визначення джерела та призначення даних, що передаються через мережу.

3. HTTP не прямо взаємодіє з фізичними протоколами, такими як Ethernet або Wi-Fi. Він використовує транспортний рівень для передачі даних через фізичні канали зв'язку[38].

3.4 Перспективи розвитку протоколів

Існує декілька видів перспектив розвитку:

По перше, це збільшення продуктивності та масштабованості: протоколи рівня застосунків повинні бути оптимізовані для обробки великих обсягів даних в реальному часі, що особливо важливо для IoT-пристроїв та хмарних обчислень. Розробка протоколів, які можуть ефективно працювати в розподілених системах, забезпечуючи при цьому високу продуктивність навіть при збільшенні кількості користувачів та пристроїв.

По – друге, це покращення безпеки: розробка нових механізмів шифрування та автентифікації для забезпечення захищеної передачі даних. Інтеграція засобів виявлення вторгнень та захисту від атак у протоколи.

По-третє, це стандартизація: забезпечення сумісності протоколів з різними операційними системами та апаратними платформами. Активне

розроблення та прийняття міжнародних стандартів, що забезпечують уніфіковані підходи до розробки та впровадження протоколів.

По-четверте, це енергоефективність: оптимізація протоколів для зменшення енергоспоживання, що особливо важливо для IoT-пристроїв з обмеженими ресурсами. Розробка протоколів, які можуть ефективно використовувати наявні ресурси, зменшуючи затримки та покращуючи швидкість передачі даних.

По-п’яте, це інтелектуальні функції та аналітика: використання алгоритмів машинного навчання та штучного інтелекту для покращення ефективності та адаптивності протоколів. Вбудовані функції аналітики, що дозволяють протоколам автоматично адаптуватися до змін умов мережі та навантаження.

Останнім пунктом можна представити нові сфери застосування: Розробка протоколів, які зможуть ефективно використовувати можливості нових поколінь мобільних мереж, таких як 5G [39]. А також спеціалізовані протоколи для конкретних галузей застосування IoT, таких як промисловий інтернет речей (IIoT), розумні міста, охорона здоров'я та ін.

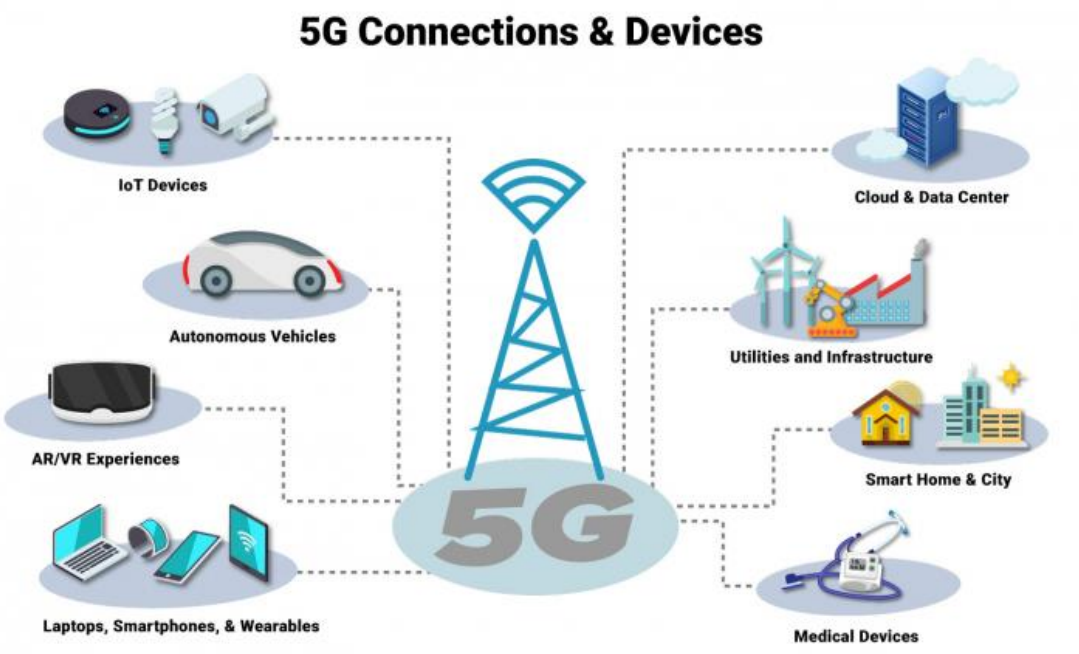

Рисунок 3.15 – 5G Connections and Devices

3.5 Висновки до розділу 3

Отже, розгляд протоколів рівня застосунків у мережах IoT показав їх ключову роль у забезпеченні ефективного обміну даними та взаємодії між різними пристроями та системами. Приклади застосування протоколів у різних галузях демонструють їхню універсальність та важливість для сучасних технологій. Хмарна обробка значно розширює можливості IoT, забезпечуючи масштабованість та високу продуктивність. Взаємодія з протоколами інших рівнів забезпечує цілісність та надійність IoT-систем. Перспективи розвитку протоколів вказують на подальше покращення їхніх можливостей та безпеки, що сприятиме широкому впровадженню IoT у різні сфери життя.

ВИСНОВКИ

Під час написання дипломної роботи було розглянуто та проаналізовано особливості транспортних мереж та мереж доступу. Варто зазначити, що транспортні мережі забезпечують високошвидкісну передачу даних на великі відстані. Мережі доступу – підключення кінцевих користувачів до основної мережі.

Розвиток мереж IoT значно змінив сучасну телекомунікаційну інфраструктуру, сприяючи новим підходам до збору, обробки та передачі даних. Мережі IoT інтегруються в існуючу телекомунікаційну інфраструктуру, використовуючи як провідні, так і бездротові технології доступу.

Архітектура мережі IoT складається з кількох рівнів, кожен з яких виконує специфічні функції для забезпечення надійного збору, обробки та передачі даних.

Під час розгляду протоколів рівня застосунків варто зазначити, що:

MQTT є легким протоколом обміну повідомленнями, який використовує модель публікації/підписки. Протокол розроблений для пристроїв з обмеженими ресурсами та мереж з низькою пропускною здатністю, що робить його ідеальним для IoT.

DDS є стандартом для обміну даними в реальному часі з високою продуктивністю та надійністю. Використовує модель публікації/підписки та підтримує широкий спектр якості обслуговування (QoS) параметрів.

CoAP розроблений для роботи в обмежених умовах, таких як низька пропускна здатність мережі та обмежені ресурси пристроїв. Протокол використовує модель клієнт/сервер.

AMQP забезпечує надійний обмін повідомленнями між різними додатками та системами. Протокол підтримує різні моделі маршрутизації та забезпечує високу надійність доставки повідомлень.

HTTP є основним протоколом для обміну даними в Інтернеті та використовується для взаємодії веб-додатків. Протокол використовує модель клієнт/сервер та забезпечує підтримку широкого спектра додатків через простоту.

Отже, протоколи рівня застосунків відіграють важливу роль у розвитку та впровадженні мереж IoT. Вони забезпечують ефективний та надійний обмін даними між пристроями, підтримують інтеграцію з хмарними сервісами та гарантують безпеку переданих даних. Вибір протоколу залежить від вимог конкретного застосування, але у кожному випадку вони забезпечують критичні можливості для сучасних рішень. Подальший розвиток цих протоколів сприятиме ще ширшому впровадженню IoT у різні галузі та сфери діяльності, забезпечуючи нові можливості для підвищення ефективності та інновацій.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. П.П. Воробієнко, Л.А. Нікітюк, П.І. Резніченко. «Телекомунікаційні та інформаційні мережі». - 2010. [Електронний ресурс] URL: https://duikt.edu.ua/uploads/l_472_12078122.pdf

2. Рішення IoT: розширення можливостей майбутнього з революційним впливом. [Електронний ресурс] URL: https://www.dusuniot.com/uk/blog/iot-solution-empowering-the-future-with-revolutionary-impact/

3. Лекція: Загальні принцип побудови та архітектура IoT. [Електронний ресурс].URL:https://learn.ztu.edu.ua/pluginfile.php/205968/mod_resource/content/2/%D0%9B%D0%B5%D0%BA_1.pdf

4. Стаття: Інтернет речей: мережева архітектура та архітектура безпеки. [Електронний ресурс]. URL: https://www.bizmaster.xyz/2020/12/internet-rechei-merezheva-arkhitektura-ta-arkhitektura-bezpeky.html

5. Pratikkumar Desai, Amit Sheth, Pramod Anantharam. Semantic Gateway as a Service Architecture for IoT Interoperability. URL:https://www.researchgate.net/publication/267157426_Semantic_Gateway_as_a_Service_Architecture_for_IoT_Interoperability

6. Eman M Abou-Nassar, Abdullah M. Iliyasu, Passent Elkafrawy, Oh-Young Song, Ali Kashif Bashir, Ahmed A. Abd El-Latif (2020). DITrust Chain: Towards Blockchain-Based Trust Models for Sustainable Healthcare IoT Systems. URL: https://www.researchgate.net/publication/341836087_DITrust_Chain_Towards_Blockchain-Based_Trust_Models_for_Sustainable_Healthcare_IoT_Systems

7. Стаття: Домашня автоматизація за допомогою IoT: як швидко створити рішення для розумного дому
URL: https://www.dusuniot.com/uk/blog/home-automation-using-iot/

8. Стаття: Smart Cities Development with 5G, IoT and Cloud Computing. URL: https://wh516x.medium.com/smart-cities-development-with-5g-iot-and-cloud-computing-51010dd02a82

9. Офіційний сайт MQTT. URL: https://mqtt.org/

10. Стаття: Introducing the MQTT Protocol – MQTT Essentials: Part 1 URL: https://www.hivemq.com/blog/mqtt-essentials-part-1-introducing-mqtt/

11. І. В. Стрелковська, Р. В. Золотухін, Т. І. Григор'єва. «ПОРІВНЯЛЬНИЙ АНАЛІЗ ПРОТОКОЛІВ ПЕРЕДАЧІ ДАНИХ АВТОМАТИЗОВАНИХ СИСТЕМ УПРАВЛІННЯ В НИЗЬКОШВИДКІСНИХ МЕРЕЖАХ ЗВ'ЯЗКУ». URL: http://www.sci-notes.mgu.od.ua/archive/v36/11.pdf

12. What is DDS. [Електронний ресурс] URL: https://www.dds-foundation.org/what-is-dds-3/

13. DDS Security Version 1.1 URL: https://www.omg.org/spec/DDS-SECURITY/1.1/PDF

14. OMG & IIOT STANDARDS. [Електронний ресурс] URL: https://www.omg.org/hot-topics/iot-standards.htm

15. CoAP протокол. [Електронний ресурс] URL: https://itmaster.biz.ua/directory/standarts/coap.html

16. IOT ПРОТОКОЛИ (COAP). [Електронний ресурс] URL:https://greenhouse.cv.ua/?p=598#%D0%90%D1%80%D1%85%D1%96%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0-CoAP

17. AMQP vs MQTT: Messaging protocols compared. [Електронний ресурс] URL: https://www.cloudamqp.com/blog/amqp-vs-mqtt.html

18. What is HTTP?. [Електронний ресурс] URL: https://www.cloudflare.com/learning/ddos/glossary/hypertext-transfer-protocol-http/

19. Загальні відомості про HTTP. [Електронний ресурс] URL: https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview

20. HTTP. [Електронний ресурс] URL: https://www.geeksforgeeks.org/what-is-http/

21. Як керувати датчиками розумного будинку за допомогою Google Assistant. [Електронний ресурс]
URL: https://dou.ua/forums/topic/38606/
22. Who’s using DDS? [Електронний ресурс]
URL: https://www.dds-foundation.org/who-is-using-dds-2/
23. Use Cases of CoAP. [Електронний ресурс]
URL: https://www.emqx.com/en/blog/coap-protocol#use-cases-of-coap
24. Розумний будинок. [Електронний ресурс]
URL: https://www.homedesign.in.ua/rozumnyy-budynok-iak-otsyfruvaty-sviy-budynok/
25. What is Advanced Message Queuing Protocol? [Електронний ресурс]
URL:https://nonamesecurity.com/learn/what-is-advanced-message-queuing-protocol-amqp/#:~:text=In%20AMQP%2C%20a%20message%20is,on%20the%20routing%20key%27s%20content.
26. Знайомство з розумним термостатом Nest Learning Thermostat. [Електронний ресурс] URL: https://www.smarty.ninja/hard/sensors/nest-learning-thermostat/
27. Philips Hue — Розумна система освітлення з дистанційним керуванням. [Електронний ресурс] URL: https://www.svit-lamp.ua/stati/philips-hue-rozumna-sistema-osvitlennja-z-distancijnim-keruvannjam/
28. Огляд Arlo Essential XL Spotlight. [Електронний ресурс] URL: https://hitech.mediadoma.com/uk/ogljad-arlo-essential-xl-spotlight-bilshe-oznachaie-krashhe/
29. Розумний замок August Smart Lock Pro 3rd Gen Silver. [Електронний ресурс] URL: https://secur.ua/umniy-dom/smart-zamki/umnyj-zamok-august-smart-lock-pro-3rd-gen-silver-plus-connect
30. Техніка розумного будинку: майбутнє, яке вже настало (фото). [Електронний ресурс]. URL: https://ukr.media/science/312509/#google_vignette

31. Система розумного освітлення. [Електронний ресурс] URL: https://vatra.in.ua/pro-kompaniiu/novyny/systema-rozumnoho-osvitlennia/

32. Розумний лічильник електроенергії с WiFi D103-600. [Електронний ресурс] URL: https://store.smart-maic.com/ua/p1042264810-umnyj-schetchik-elektroenergii.html

33. IoT and Cloud Computing: How Do They Work Together? [Електронний ресурс]. URL: https://www.cloudpanel.io/blog/iot-and-cloud-computing/

34. IoT і хмарні технології. [Електронний ресурс] URL: https://server-shop.ua/ua/the-internet-of-things-and-the-scope-of-its-use.html

35. Передача даних в архітектурі IIoT: MQTT. [Електронний ресурс] URL: http://edu.asu.in.ua/mod/book/tool/print/index.php?id=116

36. Тема 14. Протоколи Інтернету речей. [Електронний ресурс] URL: https://e-tk.lntu.edu.ua/pluginfile.php/20331/mod_resource/content/0/%D0%A2%D0%B5%D0%BC%D0%B0%2014.%20%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB%D0%B8%20%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82%D1%83%20%D1%80%D0%B5%D1%87%D0%B5%D0%B8%CC%86.pdf

37. RabbitMQ. Протоколи . [Електронний ресурс] URL: https://drukarnia.com.ua/articles/rabbitmq-protokoli-FfYEy

38. Типи мережевих протоколів і їх призначення (HTTP, IP, SSH, FTP, POP3, MAC). [Електронний ресурс] URL: https://deltahost.ua/ua/tipi-merezhevix-protokoliv-i-ih-priznachennya-http-ip-ssh-ftp-pop3-mac.html

39. Як розвивається 5G у світі: кейси та перспективи. [Електронний ресурс] URL: https://hub.kyivstar.ua/articles/yak-rozvivayetsya-5-g-u-sviti-kejsi-ta-perspektivi

40. Мережа 5G. [Електронний ресурс] URL: https://andro-news.com/ua/news/set-5g-kogda-poyavitsya-v-ukraine.html