Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризованих технологій
(повна назва)

Кафедра Комп'ютерно-інтегрованих технологій, автоматизації та мехатроніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

другий (магістерський)
(рівень вищої освіти)

Розробка автоматизованої системи моніторингу та візуалізації даних для кіберфізичних виробничих систем
(тема)

Виконав: студент 2 курсу, гр. КТРСм–21–1
Шалько Євгеній Вячеславович
(прізвище, ініціали)

Спеціальність 151 Автоматизація та комп'ютерно-інтегровані технології

освітньої програми Комп'ютеризовані та робототехнічні системи
(код і повна назва напряму)

Тип програми освітньо-професійна
(повна назва освітньої програми)

Керівник проф. Євсєєв В.В.
(посада, прізвище, ініціали)

Допускається до захисту
зав. кафедри _____ Невлюдов І.Ш.
(підпис) (прізвище, ініціали)

2022 р.

Харківський національний університет радіоелектроніки

Факультет	Автоматики і комп'ютеризованих технологій
Кафедра	Комп'ютерно-інтегрованих технологій, автоматизації та мехатроніки
Рівень вищої освіти	другий (магістерський)
Спеціальність	151 Автоматизація та комп`ютерно-інтегровані технології
Тип програми	освітньо-професійна
Освітня програма	Комп'ютеризовані та робототехнічні системи

(код і повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____»_____ 2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Шальку Євгенію Вячеславовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка автоматизованої системи моніторингу та візуалізації даних для кіберфізичних виробничих систем
затверджена наказом по університету від 07.11. 2022 р. № 1462 Ст.
2. Термін подання студентом роботи до екзаменаційної комісії 14.12.2022 р.
3. Вихідні дані до роботи
3.1 Система передачі: Wi-Fi
3.2 Передача даних: клієнт-серверна архітектура
3.3 Гнучкий інтерфейс
3.4 Спосіб живлення: мережа
4. Перелік питань, що потрібно опрацювати в роботі

4.1 Вступ
4.2 Аналіз сучасних систем моніторингу та візуалізацій даних на виробництві
4.3 Розробка структури та інформаційної моделі системи
4.4 Розробка автоматизованої системи моніторингу та візуалізацій
4.5 Експериментальне дослідження та перевірка працездатності системи
4.6 Висновки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів)
демонстраційний матеріал представлений у форматі презентації PowerPoint (*.ppt) – 18 с. формату А4

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Керівник (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз літератури по темі роботи	07.11.22 – 10.11.22	виконано
2	Аналіз технічного завдання	10.11.22 – 12.11.22	виконано
3	Аналіз сучасних систем стеження	12.11.22 – 14.11.22	виконано
4	Розробка апаратної частини системи стеження	14.11.22– 22.11.22	виконано
5	Розробка програми для макету	23.11.22 – 26.11.22	виконано
6	Оформлювання пояснювальної записки	27.11.22 – 7.12.22	виконано
7	Подання роботи на перевірку Інтернет-сервісом Unicheck	10.12.22	виконано
8	Оформлювання пояснювальної записки	10.12.22	виконано
9	Подання роботи на рецензію	12.12.22	виконано
10	Подання роботи на підпис зав. кафедри	13.12.22	виконано
11	Подання кваліфікаційної роботи в ЕК	14.12.22	виконано

Дата видачі завдання 7 листопада 2022 р.

Студент _____ Шалько Є.В.
(підпис) (прізвище, ініціали)

Керівник роботи _____ проф. Євсєєв В. В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 117 с., 43 рис., 9 табл., 2 дод., 36 джерел.

СИСТЕМИ МОНІТОРИНГУ, СИСТЕМИ ВІЗУАЛІЗАЦІЇ, ОДНОПЛАТНИЙ КОМП'ЮТЕР, SCADA, SCADA HMI, АЛГОРИТМ, МАКЕТ.

Мета кваліфікаційної роботи – розробка модуля SCADA/HMI для кіберфізичних виробничих систем.

Об'єкт дослідження – процес візуалізації виробничих даних для кіберфізичних систем.

Предмет дослідження – методи передачі візуалізації виробничою інформації всередині кіберфіцичної системи на базі концепції M2M.

Методи дослідження – теорія реляційних баз даних, теорія побудови інформаційних моделей.

Розроблено макет системи моніторингу та візуалізації даних для кіберфізичних виробничих систем, який має можливість збирати дані з декількох датчиків, передавати дані в середині своєї системи, аналізувати отриманні дані та виводити отриманні результати на пристрої користувача.

У роботі було розроблено структурна схема, інформаційна модель системи, архітектуру системи, логічну та фізичну модель бази даних, а також виконано розрахунок необхідне освітлення підприємства для коректної роботи системи. Розроблені алгоритм роботи та програма для виконання необхідних функцій системи. Виконано компонування пристрою та зібрано макет. Після чого проведено експериментальне дослідження по працездатності розробленої системи.

ABSTRACT

Explanatory note: 117 p., 43 pic., 9 tabl., 2 appl., 36 sources.

MONITORING SYSTEMS, VISUALIZATION SYSTEMS, SINGLE BOARD COMPUTER, SCADA, SCADA HMI, ALGORITHM, LAYOUT.

The purpose of the qualification work is to develop a monitoring and data visualization system for cyber-physical production systems.

The object of research is the process of visualization of production data for cyber-physical systems.

The subject of the study is the methods of transmission of visualization of production information within a cyber-physical system based on the M2M concept.

Research methods – the theory of relational databases, the theory of building information models.

A model of a monitoring and data visualization system for cyber-physical production systems has been developed, which has the ability to collect data from several sensors, transmit data within its system, analyze the received data, and display the received results on the user's device.

In the work, the structural diagram, information model of the system, system architecture, logical and physical model of the database were developed, as well as the calculation of the necessary lighting of the enterprise for the correct operation of the system was performed. The work algorithm and the program for performing the necessary functions of the system have been developed. The layout of the device is done and the layout is assembled. After that, an experimental study was conducted on the performance of the developed system.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ПЛК – програмований логічний контролер;

IIoT – промисловий Інтернет речей (англ. Industrial Internet of Things);

CPPS – кіберфізичні виробничі системи (англ. Cyber Physical Production Systems);

M2M – машино-машинна взаємодія (англ. Machine-to-Machine, Mobile-to-Machine, Machine-to-Mobile);

HMI – людино-машинний інтерфейс (англ. Human–machine interface);

SCADA – системи диспетчеризації, керування та збору даних (англ. Supervisory Control And Data Acquisition);

RTU – віддалений термінал (англ. Remote Terminal Unit);

MTU – диспетчерський пункт управління (англ. Master Terminal Unit);

SBC – одноплатний комп'ютер (англ. Single-Board Computer);

IDE – інтегроване середовище розробки (англ.Integrated Development Environment);

JSON – нотація об’єктів JavaScript (англ.JavaScript Object Notation);

API – інтерфейс прикладного програмування (англ.Application Programming Interface).

ВСТУП

Industry 4.0 є одним із основ поточної індустріальної ери. Дана концепція стала можливою завдяки стрімкому розвитку хмарних технологій, що дозволяє спростити багато задачі, такі як управління роботою пристроїв, збиранням даних, забезпеченням інтерактивних можливостей, підтримка зв'язку між різними об'єктами системи та багатьма іншими процесами.

Впровадження концепції Industry 4.0 сприяє значному підвищенню значущості кібернетичного аспекту виробничого процесу, що дозволяє реалізовувати нові підходи до реалізації виробництва як синтезу кібернетичних та фізичних складових – кібер-фізичні виробничі системи управління (CPPS).

Одною з основних проблем які потрібно вирішити в ході впровадження кіберфізичних виробничих систем це моніторинг виробничих процесів з додавання можливості візуалізації даних процесів. Моніторинг продуктивності та стану обладнання завжди був невід'ємною частиною інформаційних систем, що використовуються в промисловості для підвищення ефективності та мінімізації незапланованих простоїв.

На сьогоднішній день підприємства побудовані на концепції Industry 4.0 з використання всіх можливостей кібер-фізичні виробничі системи є дуже рідким явищем. Що обумовлено складністю та дорожнечею побудови подібних систем. Саме тому розробка автоматизованої системи яка може взаємодіяти з великими системами на відстані без втрат даних та у real-time, є однією з головних задач для кібер-фізичних систем. З цього можна зробити висновок, що розробка автоматизованої системи моніторингу та візуалізації даних для кіберфізичних виробничих систем, яка проводиться у даній роботі є актуальною задачею.

Мета кваліфікаційної роботи – розробка модуля SCADA/HMI для кіберфізичних виробничих систем.

Об’єкт дослідження – процес візуалізації виробничих даних для кіберфізичних систем.

Предмет дослідження – методи передачі візуалізації виробничою інформації всередині кіберфіцичної системи на базі концепції M2M.

Методи дослідження – теорія реляційних баз даних, теорія побудови інформаційних моделей.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– провести аналіз сучасних систем моніторингу та візуалізацій даних на виробництві;

– провести аналіз існуючих SCADA та SCADA/HMI;

– провести розробку структури та інформаційної моделі системи;

– вибрати апаратні модулі для макету;

– розробити алгоритм і програму для виконання необхідних функцій пристрою;

– зібрати макет пристрою;

– провести експеримент та проаналізувати отримані дані;

– оформити кваліфікаційна роботу відповідно з [1], [2].

1 АНАЛІЗ СУЧАСНИХ СИСТЕМ МОНІТОРИНГУ ТА ВІЗУАЛІЗАЦІЙ ДАНИХ НА ВИРОБНИЦТВІ

1.1 Аналіз Industry 4.0 та роль кібер-фізичних виробничих систем

Тенденції розвитку сучасного виробництва ведуть до збільшення обсягів інформації, підвищення вимог до її точності і своєчасного подання для аналізу і ухвалення рішень в режимах реального часу, тобто з кожним роком стає все складнішим і складнішим. Відповідно до цього зараз відбувається перегляд багатьох підходів до використання високих технологій та їх ролі в різних сферах діяльності людини, це призводить до виникнення потреби змінення підходів до промислових технологій та способів роботи з ними.

Через подібні тенденції провідні країни в галузі інноваційних технологій в промисловості запропонували нову концепцію стратегії цифрової революції – Industry 4.0 [3].

Концепції Industry 4.0 є однією з головних вимог на будь якому виробництві. Але основними напрямками у якій використовується дана концепція це виробництво складних високотехнологічних виробів та компонентів для них, наприклад мікрочипів або складних високотехнологічних пристроїв, в які інтегровані елементи технології Industrial Internet of Things (IIoT) [3]. Ця концепція вимагає впровадження повної автоматизації виробничих процесів, при чому управління ними буде здійснюватися в реальному часі з мінімальним втручанням ззовні.

Створення виробничого циклу високотехнологічних пристроїв у сучасних умовах неможливе без використання та впровадження систем автоматизації на всіх рівнях виробництва і його подальшою експлуатацією.

Для створення подібного циклу та рівня інтеграції даної концепції необхідно використовувати комбінацію з інформаційно-комунікаційних технологій (ICT) [4], операційних технологій (OT) і кіберфізичних

виробничих систем (CPPS) на всіх етапах виробництва високотехнологічної продукції [3].

Схематично виробництво з впровадженим Industry 4.0 можна показати, як на рис. 1.1.

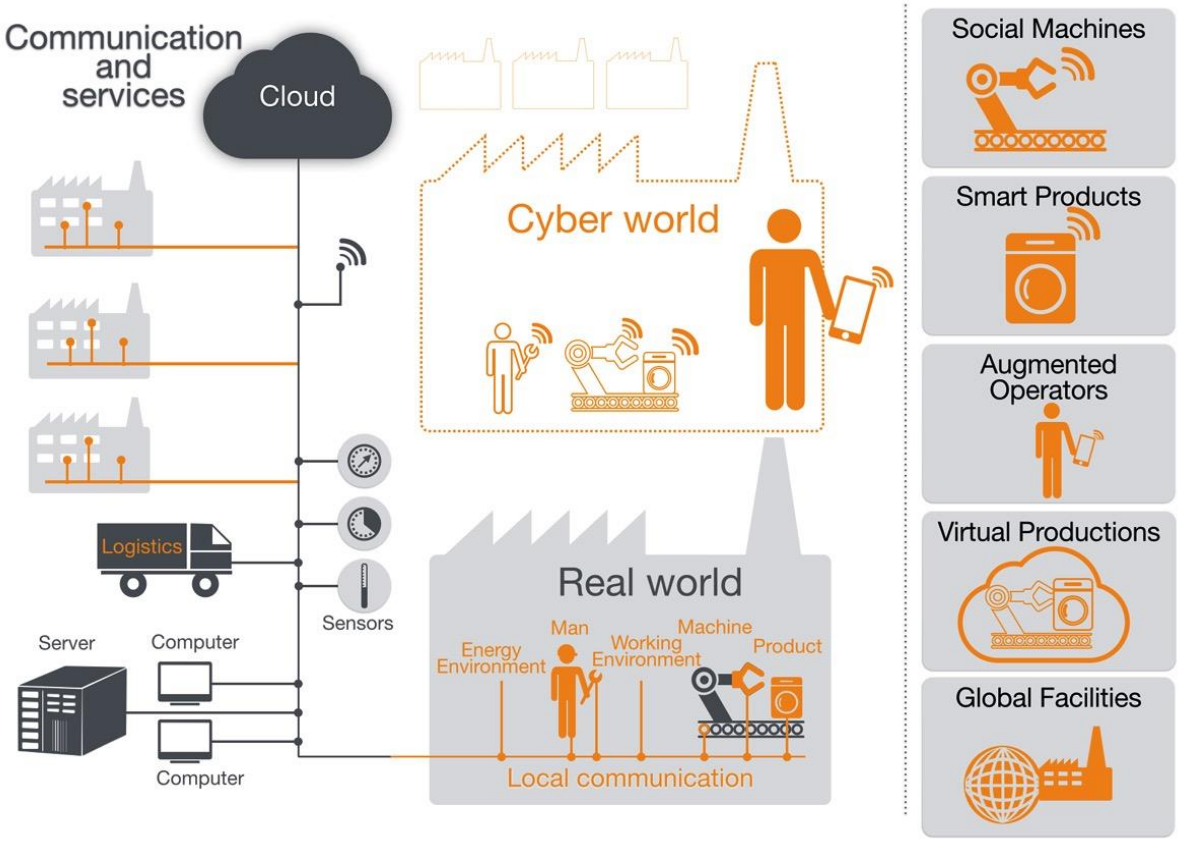


Рисунок 1.1 – Схематичне представлення компонентів Industry 4.0 [5]

Одною з ключових технологій, які впроваджуються з концепцією Industry 4.0 є CPPS, суттю якої є створення віртуального двійника реального виробництва, який дозволяє збирати дані про фізичні процеси з мехатронних пристроїв (датчиків) та керувати процесами у разі необхідності, приймати децентралізовані рішення різних завдань, що у поєднанні з програмною (кібернетичною) обробкою даних про технологічні процеси та їхвізуалізації (HMI) дозволяє створити розподілені системи для прийняття рішення.

З інтеграцією кібернетичної складової до системи роль людини

зводиться до керування процесами на фізичному рівні через кібернетичну систему. Але не зважаючи на значно меншу роль людини, дана система дозволяє: автоматично контролювати ТП на фізичному рівні, аналізувати та приймати рішення у масштабі реального часу, накопичувати великий масив технологічних даних. За рахунок цього відбувається вдосконалення керування виробничими процесами, де основною функцією людини є моніторинг та керування експертними даними.

Кіберфізична система є складною розподіленою системою, керованою або контрольованою комп'ютерними алгоритмами, характерною рисою такої системи є тісне інтегрування з Інтернет та його користувачами. Технологічною основою кіберфізичних систем є технологія інтернет-речей (Internet of Things, або IoT). У системах CPPS фізичні та програмні компоненти тісно взаємопов'язані.

З кожним роком підвищується рівень складності завдань управління, що зумовлює застосування принципово нових методів та систем управління. Кібер-фізичні системи дозволяють здійснюють обчислювальні процедури у своїй розподіленій структурі, які включають «розумні вузли» з можливістю реконфігурування потоків мережі залежно від умов. Тобто кіберфізичні системи дають можливість інтелектуальної обробки та реконфігурації потоків за рахунок інтелектуального управління. CPPS застосовує трансдисциплінарні підходи, поєднуючи теорію кібернетики, мехатроніки, проєктування та науки про процеси. Управління процесом часто називають вбудованими системами (embedded systems). Дані системи в основному спрямовані на розвиток обчислювальних елементів та меншою мірою на інтенсивному зв'язку між обчислювальними та фізичними елементами. CPPS та технологію «Інтернет речей», використовують ту саму базову архітектуру. Однак CPPS є більш високою комбінацією та координацією між фізичними та обчислювальними елементами [6].

Для впровадження технології Industry 4.0 в структуру виробництва, засновану на елементах CPPS, що дає можливість реалізації децентралізованого автономного керування складними організаційно-технічними виробничими об'єктами та пов’язаними з ними процесами пропонується згрупувати їх у 3 кластера:

– моніторинг процесу, тобто збір і обробка даних;

– міжмашинні зв'язки (M2M);

– взаємодія людини з машиною (HMI).

До кластеру моніторингу можна віднести безліч рішення, проте основним є використання систем контролю за робочим простором за допомогою різноманітних датчиків і виконавчих механізмів в мехатронних пристроях для взаємодії з фізичним світом. В залежності від підприємства, система контролю та етапу транспортування використовуються ті чи інші датчики такі як датчики відстані, датчики наявності, датчики положення, датчики наближення, датчики орієнтації, датчики розташування, індуктивні датчики, ємнісні датчики і багато інших [4]. Це, у поєднанні з підходами технології хмарних обчислень, дає можливість застосовувати інтелектуальні об'єкти CPPS оснащені мікроелектронікою, датчиками, модулями зв'язку і обробки [7].

Результатом цього процеси виробництва набувають форму та ознаки базового інтелекту. За допомогою IIoT на виробництвах створюється середовище для об’єднання подібних інтелектуальних об'єктів та процесів в єдиний інформаційний простір. Після чого отримані дані процесів, на основі інтелектуальних об'єктів, зберігаються на платформах даних в якості бази даних (БД) звідки дані можна отримати у будь-який момент для різноманітних аналітичних додатків. У більшості випадків аналітичні додатки, залежать безпосередньо від датчиків і виконавчих механізмів і генерують дані інтелектуальних пристроїві.

Зібрані дані о процесах та об’єктах дозволяють аналізувати величезну кількість статистичних даних про технологічні процеси, наприклад дозволяє визначити найбільш нестабільні параметри, що дасть можливість приділити більше увагу цьому етапу та уникнути зниження якості в межах встановленого діапазону [8–12]. Але збирання та аналіз великої кількості даних традиційним способом займають багато часу, що повністю виключає можливість обробки і прийняття рішень в режимі реального часу.

Ця проблема вирішується за допомогою використання різних засобів автоматизації, наприклад ПЛК чи SCADA, в залежності від рівня, на якому вони застосовуються у вертикальній структурі CPPS.

Кластер M2M дає можливість забезпечити автоадаптивне управління взаємопов'язаними машинами та обладнанням без участі людини [13,14]. Впровадження технології M2M, дозволяє відстежувати параметри і стану компонентів пристроїв, з'єднаних за допомогою дротових або бездротових технологій для обміну даними, як в двосторонньому, так і в односторонньому порядку. Крім того дана технологія поєднує в собі підхід вертикальної і горизонтальної інтеграції.

Вертикальна інтеграція використовується для об’єднання пристрою і даних на різних рівнях та дозволяє створювати зв'язки процесів на фізичному рівні з MES і ERP. Вертикальна інтеграція дозволяє здійснювати індивідуальний потік без ручного перемикання.

Горизонтальна інтеграція визначає глобальний зв'язок між обладнанням на одному рівні. Використовуючи цю інформацію виробничий процес може бути змінений автономно, відповідно до автоадаптивного плану виробництва [15].

Використовуючи цю концепцію отримано рішення, які дозволяють аналізувати велику кількість даних з датчиків, у поєднанні з можливістю екстреного втручання людини для корекції технологічних процесів під

необхідні виробничі параметри , за допомогою графічних інтерфейсів, що базуються на отриманні даних в реальному часі.

1.2 Аналіз методів розробки систем моніторингу та візуалізацій даних SCADA/HMI

SCADA (Supervisory Control And Data Acquisitio) система призначенням якої є моніторинг і диспетчерській контроль великої кількості об'єктів або одного територіально розподіленого об'єкта чи процесів.

Основною задачею SCADA-систем є збір інформації про безліч віддалених об'єктів, що надходить з пунктів контролю та відображення цієї інформації в єдиному диспетчерський центр. Крім того, задачею SCADA-систем є забезпечення довгострокового архівування даних для можливого подальшого аналізу. У обов’язки диспетчера же часто входить не тільки пасивне спостерігання за об'єктом чи процесом, але і керувати ним, реагуючи на різні ситуації.

Завдання які виконується SCADA-системами під час експлуатації:

– обмін даними з промисловими контролерами та платами введення/виведення в реальному часі через драйвери;

– опрацювання інформації в реальному часі;

– відображення інформації на екрані монітора у зрозумілій для людини формі;

– ведення бази даних реального часу із технологічною інформацією;

– аварійна сигналізація та керування тривожними повідомленнями;

– підготовка та генерування звітів про хід технологічного процесу;

Існує чотири покоління системної архітектури SCADA, які включають: перше покоління (монолітне), друге покоління (розподілене), третє покоління (мережне) і четверте покоління (архітектура SCADA на основі Інтернету

речей). Концепція Інтернету речей пов’язана з об’єднанням фізичних об’єктів із вбудованою електронікою, програмним забезпеченням, датчиками та з’єднанням для забезпечення обміну даними між цими пристроями та оператором через загальну мережу або Інтернет [16-19].

Будь-яка SCADA-система складається з трьох компонентів, а саме віддалений термінал (RTU – Remote Terminal Unit), диспетчерський пункт управління (MTU – Master Terminal Unit) та комунікаційну систему (CS – Communication System). Віддалені термінали (RTU) є компактними мікропроцесорними блоками управління, що встановлюються в декількох фізичних місцях системи. Ці пристрої схожі з програмованими логічними контролерами (ПЛК), але мають деякі відмінності. Для цього можна використовувати як примітивний датчик, що знімає інформацію з об'єкта, так і спеціалізований багатопроцесорний відмовостійкий обчислювальний комплекс, завдяки якому здійснюється обробка інформації та управління в режимі реального часу. Наприклад датчики і приводи, контролюються та контролюються через SCADA мережі за допомогою ПК або програмованого логічного контролера (ПЛК).

Більшість RTU на ринку підтримують стандартні протоколи, такі як Ethernet та RS232. Але після аналізу ринку можливо зробити висновок що незважаючи на поширеність Ethernet у теперішній час, RS232 та інші послідовні протоколи, є затребуваними в застарілих системах та системах заснованих на старих технологіях.

Диспетчерський пункт управління здійснює обробку даних та управління високого рівня, як правило, у режимі квазіреального часу. Він забезпечує людино-машинний інтерфейс (HMI). MTU може бути як одиночний комп'ютер з додатковими пристроями підключення до каналів зв'язку, так і великою обчислювальною системою або локальною мережею робочих станцій та серверів. Комунікаційна система необхідна передачі

даних з RTU на MTU і назад. В якості комунікаційна система може використовувати такі канали передачі даних: виділені лінії, радіомережі, аналогові телефонні лінії, ISDN мережі, стільникові мережі GSM (GPRS). Найчастіше пристрої підключаються до кількох мереж для забезпечення надійності передачі.

У багатьох випадках виробництва також мають спеціальний контроль центру для екранування всього процесу виробництва продукту. Центр управління зазвичай розташовані в окремій фізичній частині фабрики та зазвичай має передові обчислювальні та комунікаційні засоби.

Установка систем SCADA здійснюється в 7 етапів: вибір стандартів (SCADA System Standarts), проєктування (Design), системне програмування (System Development), забезпечення складання обладнання (Hardware Build Fabrication), встановлення (Installation), перевірка та введення в експлуатацію (Comissioning) та управління (Operate).

Сучасні центри управління мають сервери даних Людина-Машина інтерфейсу (HMI) та інші сервери для допомоги операторам в загальному управлінні заводською мережею. Це Мережа SCADA зазвичай підключається до зовнішньої корпоративної мережу та/або Інтернет через спеціалізовані шлюзи. Шлюзи забезпечують інтерфейс між зовнішньою мережею на основі IP та SCADA на основі протоколу fieldbus мережі на заводі. Шлюз забезпечує механізми перетворення протоколів для забезпечення зв'язку між двома різними мережами. Він також забезпечує кеш механізми для об’єктів даних, якими між собою обмінюються мережі, щоб покращити продуктивність шлюзу. Типовий приклад SCADA мережа показана на рис. 1.2 [20].

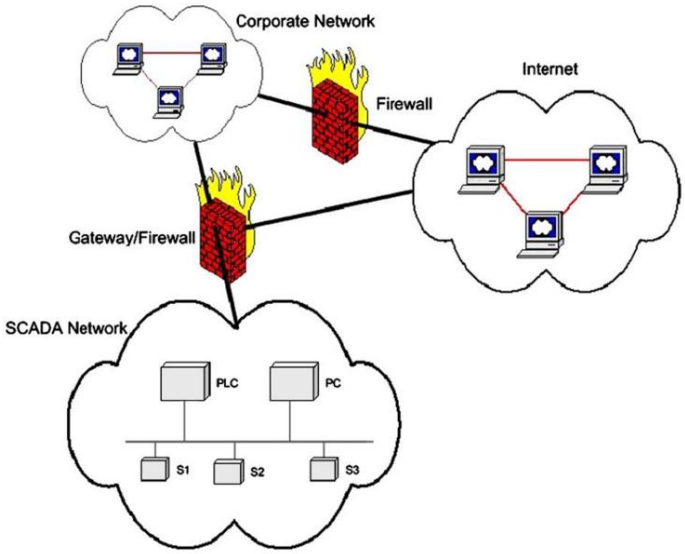


Рисунок 1.2 – Типова архітектура мереж SCADA

HMI/SCADA — це категорія програмної архітектури системи управління, що використовує мережеві дані для надання графічного інтерфейсу користувача, завдяки якому оператор контролює продуктивність багатьох одиниць обладнання і видавати технологічні команди та налаштування. Цей результат досягається за допомогою спеціального екрана, мобільного пристрою або будь-якого ПК, підключеного до мережі керування через веб-браузер.

Подібні архітектури дозволяють приймати зважені рішення для швидкого реагування, тобто покращити ситуаційну обізнаність, мобільність візуалізації у будь-який час та в будь-якому місці, а також управління важливим обладнанням, що забезпечує централізоване представлення операцій. Перетворення оперативних даних на аналітику, яку потім можна використовувати для оптимізації процесів, є однією з головних переваг подібних систем. По суті, автоматизація стає основою стратегії оцифрування компанії.

Системи, що використовують HMI/SCADA збирає дані з віддалених терміналів, програмованих логічних контролерів та інших пристроїв

керування. Ці дані надаються оператору за допомогою людино-машинного інтерфейсу (HMI). HMI дозволяє оператору бачити, що відбувається на підприємстві в режимі реального часу, включаючи мнемосхеми, що настроюються, сигнали тривоги, тренди та інші, щоб приймати рішення з налаштування будь-яких елементів управління або налаштувань машини.

Програмну архітектуру HMI/SCADA використовується в поєднанні з іншими технологіями для підвищення продуктивності, наприклад, до архіватора даних, для забезпечення аналізу тенденцій та інших аналізів на тривалому проміжку часу [21].

Сучасні HMI/SCADA, включаючи архів даних та технології централізованої візуалізації, дозволяють досягти високопродуктивного середовища розробки та візуалізації, що дозволяє оптимізувати роботу заводу, що підтримується швидшою розробкою, демократизацією інструментів та можливостей на всьому заводі, підвищенням продуктивності, зниженням витрат, зміною мислення серед співробітників. та культурою безперервної роботи [16].

Окрім міркувань продуктивності, вимоги до дизайну мережі SCADA також формуються умовами роботи мережі. Ці умови впливають на топологію мережі та на мережевий протокол. Отримані мережі SCADA мають певні унікальні характеристики. Наприклад, більшість термінальних пристроїв у мережах fieldbus є спеціальними вбудовані обчислювальні системи з обмеженими обчислювальними можливостями та функціональністю. На відміну від багатолюдних корпоративних офісних мереж, багато галузевих додатків SCADA мереж, таких як розподіл електроенергії, зазвичай рідкісні, але географічно розподілені. Обидва великі утиліти і заводські мережі часто піддаються широким коливання температури, електромагнітне випромінювання та навіть просте скупчення великої кількості пилу. Всі з ці умови збільшують шум у мережі, а також

зменшити термін служби проводів. Специфікації для фізичний рівень мережі повинен витримувати такі суворі умови та керуйте шумом у мережі.

Типові комунікації в мережі SCADA включають керуючі повідомлення, якими обмінюються головний і підлеглий пристроїв. Головний пристрій – це пристрій, який може керувати роботою іншого пристрою. ПК або ПЛК є прикладом головного пристрій. Веденим пристроєм зазвичай є простий датчик або виконавчий механізм який може посилати повідомлення на командний пристрій і переносити дії за командою головного пристрою. однак, мережевий протокол також повинен надавати функції для зв’язку між пристроями fieldbus, які хочуть спілкуватися як одно рівневі. Щоб задовольнити ці вимоги, протоколи такі як PROFIBUS мають гібридну модель зв’язку, яка включає модель однорангового зв’язку між головними пристроями та зв’язок клієнт-сервер модель між панами і рабами. Зв'язок між пристроями також може бути асиметричним. Наприклад, повідомлення, надіслані від підлеглого до головні, як правило, набагато більші за надіслані повідомлення від головного до підлеглого. Так як багато пристрої використовують спільну шину, протокол повинен мати особливості для призначення пріоритетів повідомленням. Це допомагає розрізнити критичні та некритичні повідомлення. Наприклад, тривожне повідомлення про можливе порушення техніки безпеки слід приймати пріоритет над звичайним повідомленням про оновлення даних. SCADA мережеві протоколи також повинні забезпечувати певний ступінь доставки впевненість і стабільність. Багато заводських процесів потребують зв’язку між fieldbus пристроями в реальному часі. Мережа протокол повинен мати функції, які не тільки гарантують, що критичні повідомлення доставляються, але вони доставляються в межах часових обмежень.

Відповідно до стандарту AGA-12 Американської газової асоціації існує від 150 до 200 протоколів SCADA. Більшість із цих протоколів були власними

стандартами, розробленими окремими компаніями. З роками галузь перейшла до прийняття загальних відкритих стандартних протоколів. Навіть з відкритими протоколами існує велика кількість різних професійних організацій, які змагаються за те, щоб отримати більше визнання свого стандарту протоколу в галузі. У таблиці 1.1 наведено кілька найбільш популярних і широко використовуваних протоколів [20].

Таблиця 1.1 – SCADA протоколи

	Protocol	Organization/standard	Main features
1	Ethernet/IP (Industrial Protocol)	Open DeviceNet Vendors Association (ODVA) (www.odva.org)	Об'єктно-орієнтований, протокольний; забезпечує взаємодію в мережах Ethernet і fieldbus
2	DeviceNet	Open DeviceNet Vendors Association (ODVA) (www.odva.org)	Належить до сімейства CIP (Control and Information Protocol); Протокол CAN визначає рівні 1 і 2; решта визначаються DeviceNet і CIP
3	ControlNet	ControlNet International (www.controlnet.org)	Належить до того самого сімейства CIP (Control and Information Protocol); новий фізичний рівень з вищою швидкістю, суворим детермінізмом і повторюваністю з більшим діапазоном
4	PROFIBUS	Type 3 protocol of IEC Standard 11674 and 61158	3-рівнева модель OSI; має розширення для функцій безпеки; Версія ProfiNet забезпечує сумісність з Ethernet
5	MODBUS TCP/IP	MODBUS-IDA (www.modbus.org)	Інкапсулює пакети fieldbus TCP; намагаючись стати стандартом IETF
6	DNP3	(IEC) Technical Committee 57, Working Group 03 standard	Він також базується на 3-рівневій моделі OSI
7	Foundation Fieldbus	The Fieldbus Foundation/open standard protocol	Містить багато функцій безпеки, які роблять його хорошим кандидатом для критично важливих додатків

1.3 Аналіз архітектури та основних етапів при розробці SCADA

Архітектура систем SCADA представляє собою багато рівневу систему з великою кількістю пристроїв та сторонніх систем, що взаємозв’язані між собою. Загальна архітектура систем SCADA представлена на рисунку 1.3.

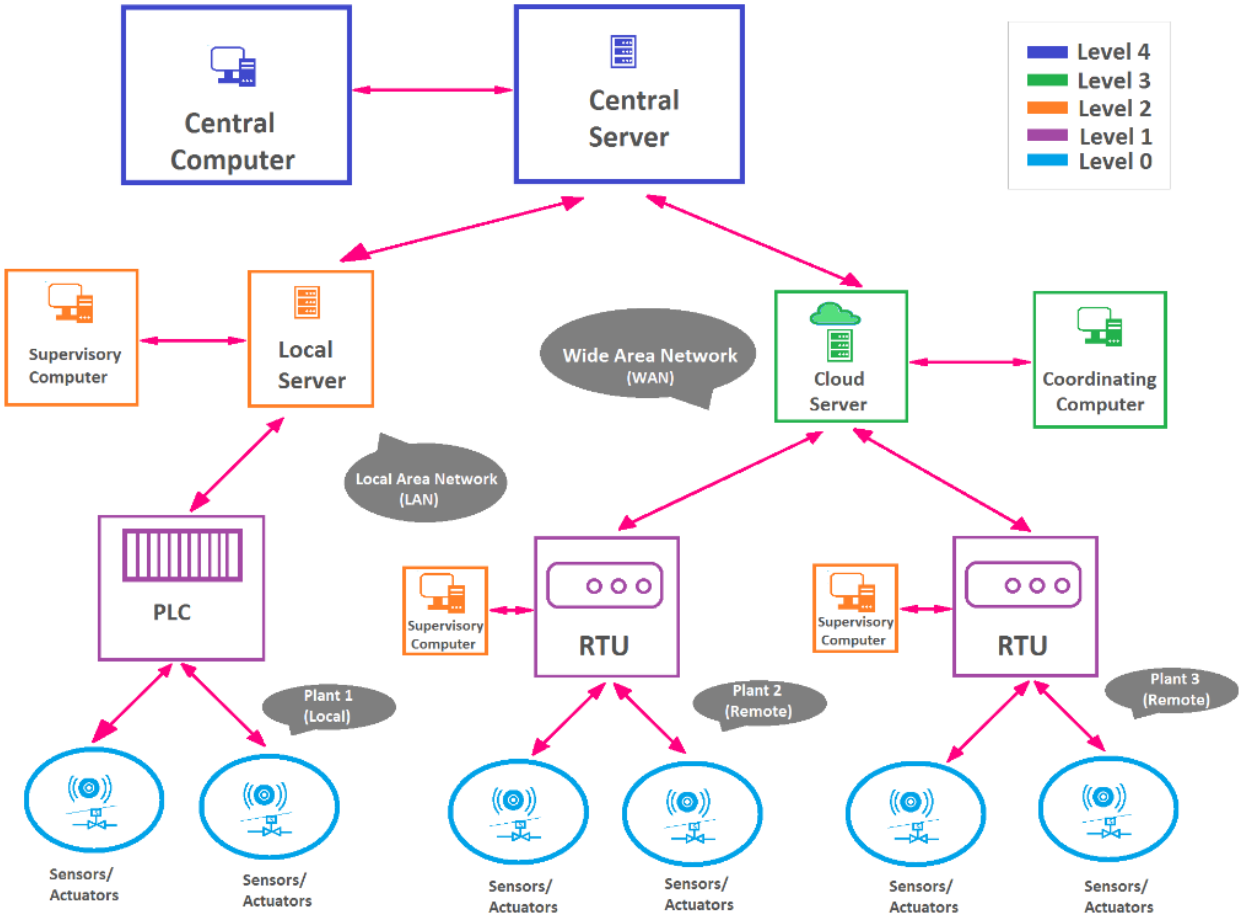


Рисунок 1.3 – Архітектура системи SCADA [22]

Згідно цієї схеми всю систему SCADA можна поділити на п’ять рівнів. Їх можна визначити як:

– рівень 0, до якого відносяться датчики та приводи;

– рівень 1, до якого відносяться пристрої програмування;

– рівень 2, до якого відносяться пристрої локального керування та HMI;

– рівень 3, до якого відносяться пристрою та сервери для координація

роботи системи;

– рівень 4, до якого відносяться центр управління та контролю системи.

Рівень 0. Наземні пристрої, які фактично взаємодіють із фізичним середовищем або працюють як техніки в системі спостереження. До цього рівня входять різні типи датчиків і приводів. Датчик — це пристрій, який може сприймати фізичні зміни навколо нього та генерувати відповідні електричні чи електронні сигнали.

Рівень 1. До цього рівня належать різні типи пристроїв програмування, наприклад програмований логічний контролер (ПЛК), дистанційний термінальний пристрій (RTU). Це пристрої, які безпосередньо керують пристроями на рівні землі, такими як датчики та виконавчі механізми. Система SCADA може бути побудована лише з локальними мережами або з комбінацією як локальних, так і глобальних мереж. ПЛК допомагає побудувати систему SCADA лише з локальною мережею, тоді як система RTU допомагає побудувати систему SCADA з глобальною мережею.

Рівень 2. До цього рівня належать комп’ютери контролю. До цих комп’ютерів під’єднані всі пристрої програмування, які керують пристроями наземного рівня. Контрольні комп’ютери – це ті, з яких програмне забезпечення SCADA починає працювати. З цих комп’ютерів надаються фактичні інструкції та команди для виконання операцій. Контролюючий комп’ютер може бути під’єднаний до певної машини або кількох однотипних машин або цілого виробничого підприємства. Цими комп’ютерами керують оператори машин, керівники заводів і технічні працівники заводу-виробника. Основними функціями цих комп’ютерів є спостереження та контроль виробництва, помилок тощо.

Рівень 3. Координаційні комп'ютери знаходяться нижче рівня. Як правило, ці комп’ютери підключені до кількох заводів. Таким чином, це може допомогти збирати дані з різних рослин з одного місця. На цьому рівні

планування виробництва, складання графіків, керування часом подій здійснюються відповідальним заводом, менеджерами тощо.

Рівень 4. Це верхній рівень системи SCADA. На цьому рівні центральний комп’ютер підключений до всіх установок і обладнання. Як правило, цим керує та контролює команда менеджерів. Тут збираються та зберігаються всі дані та інформація. Використовуючи ці дані та інформацію, вони можуть прийняти будь-яке рішення. З цього комп’ютера команда управління може бачити всі дії та операції тощо [22].

Під час проєктування SCADA-систем треба враховувати:

– обсяг даних, продуктивність, підтримка стандартних мережевих протоколів та форматів даних;

– стандартизація інтерфейсу користувача, наявність і зручність мови опису даних і процесів;

– опис пакету та експлуатаційних інструкцій;

– рівень технічної підтримки з урахуванням доступності;

– надійність використовуваної системи;

–ціна програмного продукту.

Найбільш поширеними є такі SCADA-системи:

– In Touch (Wonderware, США);

– iFix (Intellution, США);

– Genesis (Iconics Co, США);

– Citect (CI Technology, Австралія);

– Factory Link (United States Data З, США);

Критерії оцінки з позицій користувача поділяють на три групи показників: експлуатаційні, економічні та технічні характеристики.

Проєктування апаратного та програмного забезпечення для людино-машинного інтерфейсу та диспетчерського управління та збору даних (SCADA) є складним та довгим процесом. Для цього необхідно окрім

розуміння основних принципів та відповідних стандартів, також враховувати безліч супутніх факторів. До них належать стандарти компанії, можливості користувачів, рівні навичок, технологічні потреби, плани на майбутнє та інтеграція з іншими системами.

Один із перших та найважливіших етапів є визначення стандартів необхідних на виробництві. Стандарти загальні для галузі або нормативні стандарти є необхідними, однак створення власних стандартів на їх основі є основним завданням. Наявність корпоративних концепцій дозволяє організувати виробничий процес та уникнути можливих критичних поломок при неправильних рішеннях.

Для кращої роботи SCADA систем кожну систему необхідно модифікувати під потреби кожного нового виробництва, проте ця необхідність є складною для більшості постачальників даних систем. Через що можна дійти невтішного висновку, що багато систем допрацьовуються без участі початкових розробників. Така практика необхідна підтримки власних стандартів для ПЛК, протоколів, структури даних та інших елементів системи.

Під час початкової розробки проєкту необхідно розглядати систему комплексно, не обмежуючись конкретикою чи якимось вузькоспеціалізованим напрямом. Тобто при розробці слід враховувати очікувані результати роботи з системою для всіх працівників, пов'язаних із даною системою, наприклад оператори, інтегратори, супервайзери, IT-команда, яка відповідає за коректність роботи системи. Потреби кожної ланки виробництва мають бути враховані під час розробки.

Система, що розробляється, також повинна враховувати потребу, навички та досвід персоналу, який буде взаємодіяти з даною системою. Тобто, якщо на підприємстві раніше вже використовувалася якась система SCADA і її потрібно замінити або оновити, необхідно враховувати вплив

внесених змін на робочий процес співробітників, часу на навчання роботі з новою системою, а також можливого простою через внесені зміни.

Одним з нових напрямків у проєктуванні SCADA систем є використання програмного забезпечення з мобільних пристроїв. Тобто, необхідно враховувати можливість використання дизайну, що реагує на мобільні пристрої. Розробка та адаптування дизайну робочого стола до мобільного пристрою складніший процес, ніж розробка дизайн для мобільного пристрою з самого початку роботи та обліку при розробці обох версій.

Інтерфейс сучасної SCADA має бути інтуїтивно зрозумілим для подальшого успішного розвитку даної системи. Тобто необхідно уникати використання «підказок», поєднань клавіш, клацань правою кнопкою миші у певних частинах екрану тощо. Подібні елементи приховують частину функціоналу вашої системи, що робить її менш зрозумілою у процесі експлуатації.

У процесі розробки також необхідно враховувати можливість системи інтеграції коїться з іншими системами. Цей етап проєктування частково залежить від попередніх етапів. Так, наприклад, у разі використання підприємством відкритих, чітких стандартів інтеграція відбувається набагато простіше. Хорошим інструментом для цього є хмарні послуги, що використовуються для зберігання даних та обміну ними між різними системами. Дуже корисно використовувати сучасні інструменти та методи, такі як інтерфейси прикладних програм (API) OPC UA, та зберігати у форматі відкритої бази даних. Можливість легкого доступу до даних також допоможе встановити взаємозв'язок між операційними технологіями та інформаційними технологіями (IT).

1.4 Аналіз існуючих SCADA/HMI та їх недоліки

Для аналізу та виявлення недоліків існуючих SCADA/HMI було створена таблицю з декількома SCADA системами, які представлені у таблиці 1.2. Після чого було проаналізовані основні недоліки цих систем у разі використання їх у кіберфізичних системах. Дані для аналізу були взяті на основі даних з [23].

Таблиця 1.2 – SCADA протоколи [23]

Назва SCADA системи	DAQFactory	Ignition SCADA	SIMATIC SCADA	COOX MESbox SCADA	VTScada
Особливості програмного забезпечення SCADA					
Управління сигналізацією	+	–	+	+	–
Архівування та зберігання	–	–	–	+	+
Агрегація даних	+	+	+	–	+
Візуалізація даних	–	–	+	+	–
Інтеграція HMI	+	+	–	+	+
Апаратна інтеграція	–	+	+	–	+
Моніторинг у реальному часі	+	–	+	+	–
Формування звітів	+	+	+	+	+
Безпека системи	+	+	+	+	+
Підтримувані пристрої	Windows	Windows Mac Web-based	Windows Mac Android iPhone/iPad Web-based	Windows Mac Android iPhone/iPad Web-based	Windows

Аналізуючи отримані дані можна зробити висновок, що незважаючи на постійні оновлення та нові версії SCADA у своїй основі усі вони ґрунтуються на технології, якою вже кілька десятків років. Через це багато SCADA мереж є дуже обмеженими у своїх можливостях цією старою технологією, через що їх використання у кіберфізичних системах викликає багато помилок при роботі та складнощів при впровадженні. Незважаючи на безліч застосувань, стара технологія SCADA утримує своїх користувачів від використання їхнього повного потенціалу або впровадження інновацій у майбутньому.

До важливих недоліків які потребують як найшвидшого рішення можна віднести такі пункти, як погана інтеграція між частинами SCADA компонентів та монолітність мереж, використання програмним забезпечення SCADA для обміну даними лише певним типом баз даних, неточність аналітики у режимі реального часу, складність розгортання та встановлення SCADA системи у виробництво, не універсальність SCADA систем.

Надійна інфраструктура мережі є основною потребою для довготривалого використання будь-якою системи, але замість того, щоб створювати програмне забезпечення з нуля з можливістю створення нової більш просунутої технології, старі SCADA-компанії збирають окремі технологічні елементи, а потім продають їх як єдиний програмний пакет. Це призводить до того, що розрізнені частини старих систем SCADA у комбінації з новими елементами не можуть працювати без збоїв, через що на подібну систему досить важко покладатися.

Можливість швидко розширити або зменшити масштаб своєї системи є однією з основних вимог для сучасних систем, але старі системи SCADA у більшості випадків являють собою монолітні системи, які нелегко розширювати. У подібних системах виникає проблема, що чим більше вона розширюється та налаштовується під виробництво, тим складнішими вона стає.

У сучасних виробництвах де усі процеси все більше залежить від даних та швидкості надходження цих даних, велика увага приділяється управлінню великими наборами даних та отримання більшої віддачі від них за допомогою аналізу. Для виконання цієї вимоги дані мають передаватися вільно між усіма елементами системи, але на багатьох підприємствах цього немає. Однією з причин є те, що більшість програмного забезпечення SCADA обмінюються даними лише з певним типом бази даних, які заздалегідь обрані для системи, але досить часто потрібно мати можливість обмінюватися даними між різними типам баз даних, ця проблема є досить частою і призводить до затримок у роботі виробництва, неточність аналізу та можливих проблем на будь-якому етапи виробництва.

Старе програмне забезпечення SCADA зазвичай погано працює з базами даних SQL, навіть незважаючи на те, що вони є типом баз даних, що найчастіше використовується. Це може залишити розрив між тимчасовими рядами та іншими корисними корпоративними даними, які зазвичай зберігаються у базах даних SQL. В результаті дані залишаються заблокованими, а не розповсюджуються по всьому підприємству, що обмежує ефективність та інновації.

У міру прискорення темпів бізнесу та виробництва отримання даних у режимі реального часу стає дедалі важливішим і необхідним на будь якому виробництві. Старе програмне забезпечення SCADA може давати можливість відображення заводських даних в режимі реального часу, але лише вивід даних о поточному статусі етапу роботи, дає досить поверхневі відомості о ефективності, коректності та стабільності процесу.

Без можливості легко збирати і аналізувати дані в режимі реального часу та історичні дані з різних джерел, практично неможливо помістити дані в контекст. Це ускладнює відстеження тенденцій, визначення того, хто що зробив і коли, відстеження продуктивності або підвищення продуктивності

протягом тривалого часу.

Застарілі технології працюють повільніше і їх складно встановлювати, це також стосується багатьох систем на основі SCADA мереж. Багато систем досі потрібно встановлювати через фізичні носії, що також сповільнює швидкість встановлення системи. І навіть при наявність програмного забезпечення, його встановлення може займає години, дні чи навіть тижні.

Проблеми та втрата часу цьому не закінчується. Додавання функціонала чи внесення змін до конфігурації системи також викликає проблеми, у багатьох SCADA системах для цього вам доводиться вимикати систему та втрачати дорогоцінний виробничий час. Проєктування з його допомогою займає багато часу, тим більше що старі версії SCADA не підтримують паралельну розробку.

Більшість програмного забезпечення SCADA призначена для роботи у певній пропрієтарній операційній системі, зазвичай розробленій Microsoft. Це піддає користувачам стресу через часті проблеми з безпекою, виправленнями та оновленнями Microsoft і робить їх уразливими щоразу, коли Microsoft припиняє підтримку однієї зі своїх операційних систем [24].

1.5 Постановка завдання дослідження

У ході проведених досліджень були виявлені недоліки існуючих систем такі, як погана інтеграція між частинами SCADA компонентів та монолітність мереж, використання програмним забезпечення SCADA для обміну даними лише певним типом баз даних, неточність аналітики у режимі реального часу, складність розгортання та встановлення SCADA системи у виробництво, не універсальність SCADA систем. Внаслідок чого використання існуючих SCADA систем не є оптимальним рішенням для кіберфізичних систем. В результаті цього розробка нових підходів до системи моніторингу та

візуалізації даних (SCADA/HMI) для кіберфізичних систем представляє собою актуальну задачу.

Мета кваліфікаційної роботи – розробка модуля SCADA/HMI для кіберфізичних виробничих систем.

Об’єкт дослідження – процес візуалізації виробничих даних для кіберфізичних систем.

Предмет дослідження – методи передачі візуалізації виробничою інформації всередині кіберфізичної системи на базі концепції M2M.

Методи дослідження – теорія реляційних баз даних, теорія побудови інформаційних моделей.

Результатом кваліфікаційної роботи повинна буди система моніторингу та візуалізації даних для кіберфізичних виробничих систем перевагами якої буде легка інтеграція нових частин SCADA, легкий обмін даними між частинами системи та простота впровадження системи у виробництві. Для досягнення поставленої мети кваліфікаційної роботи необхідно вирішити такі завдання:

– провести розробку структури та інформаційної моделі системи;

– вибрати апаратні модулі для макету;

– розробити алгоритм і програму для виконання необхідних функцій пристрою;

– зібрати макет пристрою;

– провести експеримент та проаналізувати отримані дані.

2 РОЗРОБКА СТРУКТУРИ ТА ІНФОРМАЦІЙНОЇ МОДЕЛІ СИСТЕМИ

2.1 Розробка структури системи візуалізацій SCADA/HMI

У ході кваліфікаційної роботи було розроблено структурну схему системи моніторингу та візуалізації даних для кіберфізичних виробничих систем. Ця система призначена для отриманні мережевих даних та їх основі надання графічного інтерфейсу користувачеві. На рисунку 2.1 зображена загальна система для подібних систем.

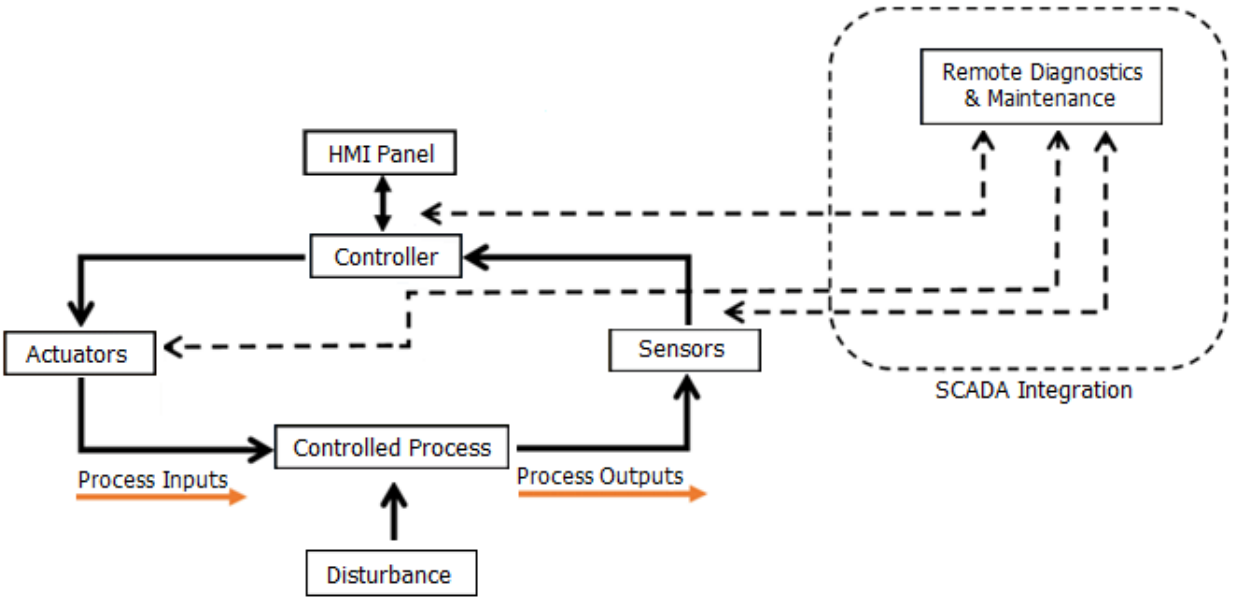


Рисунок 2.1 – Загальна структурна схема SCADA/HMI систем

Структурну схема розроблюваної системи представлена на рис. 2.2.

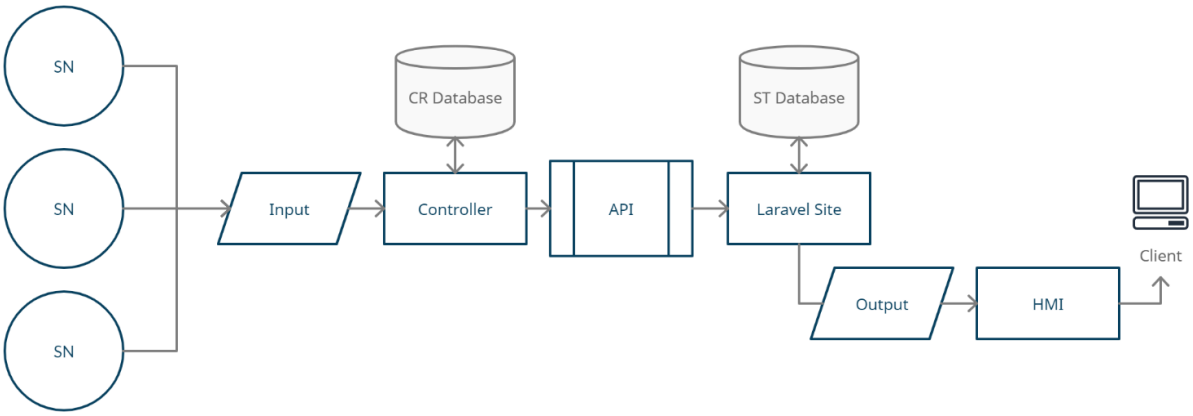


Рисунок 2.2 – Структура розроблюваної системи

На отриманій структурній схемі зображені всі основні елементи системи, такі як SN – датчики підключені до системи, Contoreller – мікрокомп'ютер Raspberry Pi, який отримує дані з датчиків зберігає їх у базу даних CR Database після чого відправляє дані через API на створену кінцеву точку у Laravel Site. Після отримання даних з Contoreller у backend частині сайту дані опрацьовуються, зберігаються у базі даних ST Databese та виводяться на екран користувача.

2.2 Розробка інформаційної моделі системи

Для розробки системи моніторингу та візуалізації даних для кіберфізичних виробничих систем було створено інформаційної моделі системи (рис. 2.3).

Інформаційна модель — система сигналів, що свідчать про динаміку об'єкта керування, умови зовнішнього середовища та стан самої системи керування. Інформаційною моделлю можуть слугувати наочні зображення, знаки, графічні моделі і комбіновані зображення.

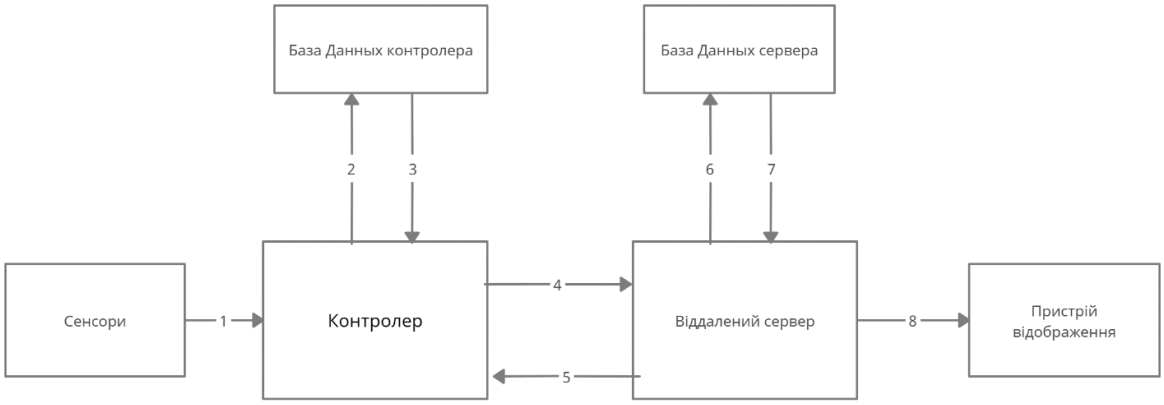


Рисунок 2.3 – Інформаційна модель системи

Розшифрування даних, які передаються у ході процесу використання системи що передаються. Стрілка під номером 1 передає об’єкт даних з датчиків до контролера, у який входить, інформація, що це за сенсор, отримані значення, локація та одиниці вимірювання. Після отримання об’єкту даних контролер записує дані у базу даних (2), крім вже існуючих даних додається час збереження та унікальний індитифікатор для цих даних. Після збереження дані з усіх об’єктів збираються (3) у один великий об’єкт та відправляться до віддаленого серверу (4). Віддалений сервер повідомляє контролер про отримання об’єкта (5), у випадку не отримання цієї відповіді контролер повторює операції 3 та 4, доки не отримає відповідь. Після розшифрування та обробки отриманого об’єкта, дані з нього завантажуються до бази даних (6) для аналізу даних з сенсорів за довгий час. Після чого дані з бази даних структуруються (7) та відображаються на пристрої користувача (8).

Після створення інформаційної моделі було розроблено архітектуру система з урахуванням ділення системи на окремі модулі, а саме модуля моніторингу та візуалізації. Кожний з модулей був проаналізований та розбитий на основні компоненти та їх складові. Архітектура зображена на рисунку 2.4.

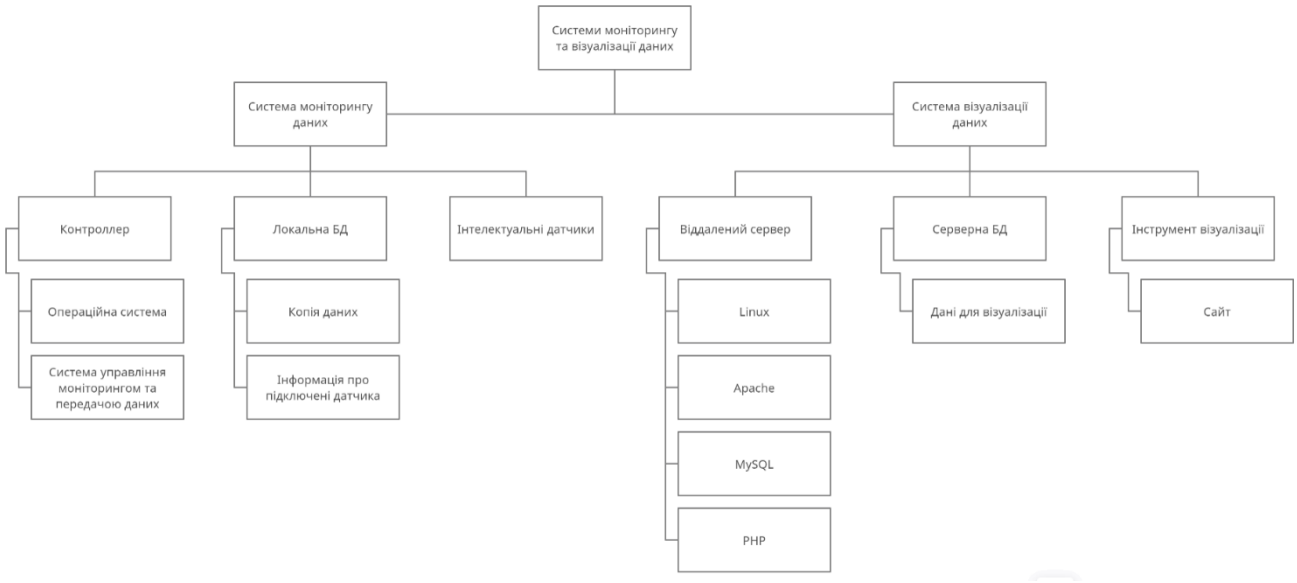


Рисунок 2.4 – Архітектура розроблюваної системи

2.3 Аналіз та вибір апаратних модулів

2.3.1 Вибір одноплатного комп'ютера

Одноплатний комп'ютер (SBC, single-board computer) представляє собою комп'ютер, на якому всі основні компоненти розміщені на одній платі. У більшості випадків одноплатні комп'ютери зустрічаються як промислові комп'ютери чи як вбудовані або комплексні офісні комп'ютери. Такий спосіб використання застосовується у системах, до яких застосовуються підвищені заходи безпеки, або у системах що повинні бути максимально компактними.

Особливістю одноплатного комп'ютера, на відміну від настільного персонального комп'ютера, є те, що подібні комп’ютери не покладаються на слоти розширення для периферійних функцій. Побудова подібних комп'ютерів заснована на використанні широкого спектру мікропроцесорів. Використовуються такі комп’ютери у випадках простих конструкції, побудовані комп'ютерними любителями, для яких часто використовують статичну оперативну пам'ять та недорогі 8-бітні або 16-бітні процесори. Інші типи, такі як блейд-сервери, виконували б аналогічно серверному комп'ютеру,

лише у більш компактному форматі [25].

Raspberry Pi Zero 2 W (рис. 2.5) є ультракомпактним одноплатним комп’ютером останнього покоління від Raspberry Pi. Цей комп’ютер придатний для використання на настільному комп’ютері загального призначення. Незважаючи на невеликий розмір і низьку вартість, Raspberry Pi Zero 2 W є повноцінною платою Raspberry Pi. Яка має 40-контактний роз’єм вводу/виводу загального призначення (GPIO), сумісний майже з усіма тими самими додатками Hardware Attached on Top (HAT), що й повнорозмірні моделі Raspberry Pi. Power over Ethernet (PoE) – сімейство Zero не має дротового Ethernet, а натомість покладається на з’єднання Wi-Fi 2,4 ГГц для підключення до Інтернету [26]. Основні характеристика Raspberry Pi Zero 2 W вказані у таблиці 2.1.

Таблиця 2.1 – Характеристика одноплатного комп'ютера Raspberry Pi Zero 2W [26]

Процесор	Broadcom BCM2710A1, quad-core 64-біт SoC
Оперативна пам'ять (RAM)	512 Мб
GPIO контактів	40
Сховище eMMC	microSD
Мережеві можливості	Wi-Fi 2,4 ГГц, 150 Мб/с
Живлення	5 В, 2,5 А
Частота процесора	1 ГГц
Робоча температура	від -20 °C до +70 °C

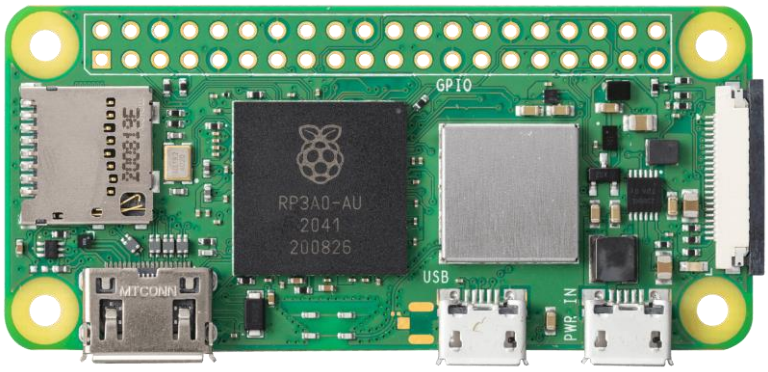


Рисунок 2.5 – Зовнішній вигляд одноплатного комп'ютера Raspberry Pi Zero 2W [26]

Raspberry Pi 4 Model B (рис. 2.6) пропонують збалансоване поєднання продуктивності, сумісності, можливостей розширення та ціни.

У порівнянні з попередніми моделями, Raspberry Pi 4 пропонує значне покращення продуктивності.

Це пов’язано з поєднанням нового процесора та графічного процесора, але значною мірою завдяки розширеній пам’яті.

Raspberry Pi 4 має переваги розвиненої екосистеми, яку компанія розробила роками, завдяки повній сумісності програмного та апаратного забезпечення з попередніми моделями.

Його популярність також означає велику підтримку.

Raspberry Pi 4 має багато можливостей підключення, включаючи високошвидкісні порти USB 3.0.

Ці комп’ютери є хорошим варіантом для обчислень загального призначення, а також ідеальний для тих, хто тільки починає свою подорож до SBC [27].

Основні характеристика Raspberry Pi 4 Model B вказані у таблиці 2.2.

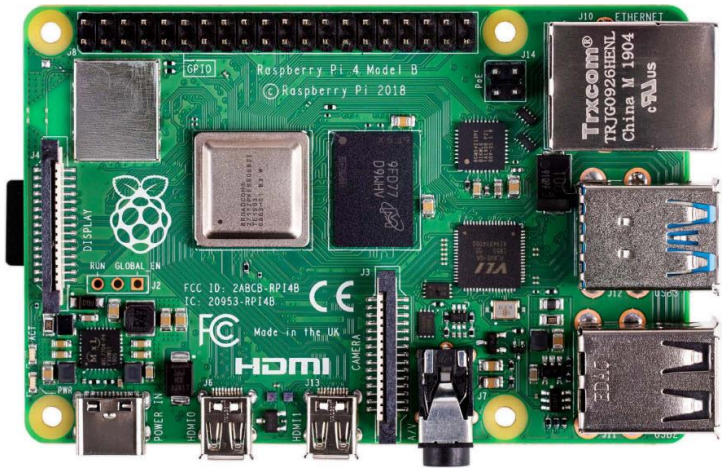


Рисунок 2.6 – Зовнішній вигляд одноплатного комп'ютера Raspberry Pi 4 Model B [27]

Таблиця 2.2 – Характеристика одноплатного комп'ютера Raspberry Pi 4 Model B [27]

Процесор	Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-біт SoC @ 1,5 ГГц
Оперативна пам'ять (RAM)	1 Гб, 2 Гб, 4 Гб і 8 Гб
GPIO контактів	40
Сховище eMMC	microSD
Мережеві можливості	Ethernet 10 / 100 / 1000 Wi-Fi 2,4 ГГц / 5,8 ГГц, 300 Мб/с
Живлення	5 В, 3 А
Частота процесора	1,5 ГГц
Робоча температура	від 0 °С до + 50 °С

LattePanda 3 Delta 864 (рис. 2.7) остання новинка в сімействі LattePanda – це повнофункціональний мініатюрний ПК, замаскований під одноплатний комп’ютер.

Процесор Intel Celeron N5105, який є серцевиною плати, забезпечує високу продуктивність при відносно низькому споживанні електроенергії, але потребує вбудований вентилятор для охолодження. Є широка сумісність з операційними системами, включаючи Windows 10 і Windows 11, а також 8 Гб оперативної пам’яті та 64 Гб вбудованої пам’яті.

Як і його попередні покоління, остання модель містить мікроконтролер Microchip ATmega32U4 як співпроцесор для робочих навантажень у режимі реального часу.

LattePanda 3 Delta поставляється в комплекті з активним вентилятором охолодження та адаптером живлення USB Type-C потужністю 45 Вт зі шнурами ЄС і США [28]. Основні характеристика LattePanda 3 Delta вказані у таблиці 2.3.

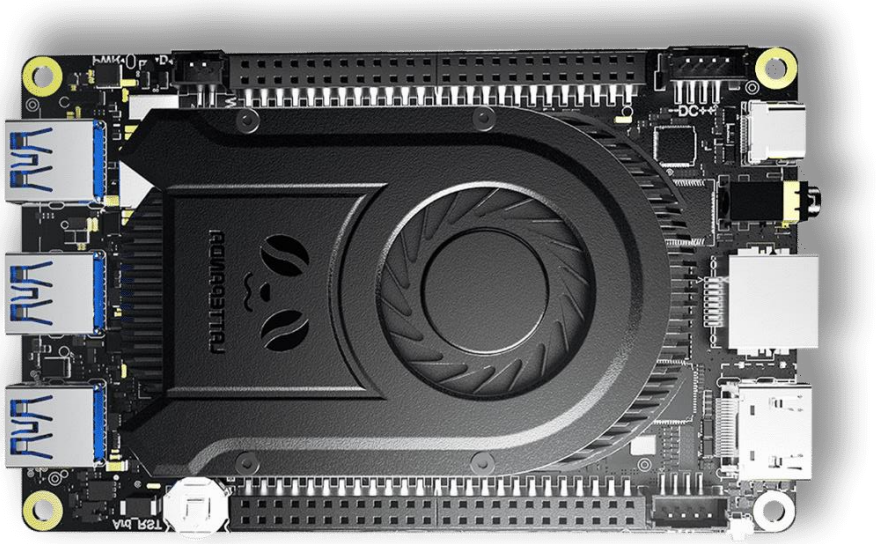


Рисунок 2.7 – Зовнішній вигляд одноплатного комп'ютера LattePanda 3 Delta [28]

Таблиця 2.3 – Характеристика одноплатного комп'ютера LattePanda 3 Delta 864 [28]

Процесор	Intel® Celeron® N5105 Quad-Core, Four-Thread
Оперативна пам'ять (RAM)	8 Гб
GPIO контактів	12 аналогових пінів 23 цифрових входів/виходів 1 UART 1 I2C 1 SPI 1 аудіороз'єм 1 порт вентилятора (4 Pin 1,25 мм PWM 5 В)
Сховище eMMC	64 Гб
Мережеві можливості	Wi-Fi 2,4 ГГц / 5 ГГц Bluetooth 5.2 Ethernet
Живлення	від 12 В до 15 В
Частота процесора	від 2,0 ГГц до 2,9 ГГц
Робоча температура	від 0 °C до + 75 °C

Проаналізувавши три варіанти одноплатних комп'ютерів від фірм Raspberry Pi та LattePanda, для досягнення мети кваліфікаційної роботи використовуємо одноплатний комп'ютер Raspberry Pi 4 Model B (рис. 2.6).

Raspberry Pi 4 Model B в порівнянні з попереднім поколінням Raspberry PI3 Model B + отримала досить багато удосконалень, такі як підвищення швидкості роботи процесора, поліпшена продуктивність мультимедійного модуля, збільшення кількості і швидкості оперативної пам'яті і оновлені

мережеві модулі, при цьому збережено рівень енергоспоживання і сумісність периферійних модулів Крім того ця плата отримала додаткове оновлення для багатьох елементів плати, серед яких підтримка одночасного підключення двох 4 К дисплеїв, 1 Гбіт/с Ethernet, 2 USB 3.0 порту.

Ключовими характеристиками даного продукту є новий, високопродуктивний 64-розрядний чотирьох ядерний процесор, підтримка двох дисплеїв з роздільною здатністю до 4 К через пару micro-HDMI портів, апаратне декодування відео, від 1 Гб до 8 Гб оперативної пам'яті, двохдіапазонна бездротова мережа на 2,4 ГГц та 5,0 ГГц, Bluetooth 5.0, Gigabit Ethernet, два порти USB 3.0 і PoE.

Для живлення моделі потрібний блок 5 В, 3 А, з можливостю використання блоки живлення на 2,5 А при споживанні підключеної периферії не більше 500 мА сумарно.

2.3.2 Датчики

Інтелектуальний датчик – це сенсорний пристрій, який здатний виконувати низку інтелектуальних функцій у рамках свого завдання чи обов'язку. Інтелектуальний датчик здатний до самотестування та пристосуватися самостійно, а також самоідентифікації. Ці датчики розуміють середовище, в яке вони потрапляють, і вони можуть керувати широким спектром умов. Інтелектуальний датчик здатний керувати своїми функціями внаслідок подразника від зовнішніх функцій. Це показує, що інтелектуальний датчик має архітектуру вдосконаленого навчання, адаптації та обробки сигналів, все в одному інтегральному ланцюзі. Для інтелектуального датчика потрібен спеціалізований апаратний апарат, який називається схемою кондиціонування сигналу, щоб контролювати та керувати собою та іншими приладами [29].

Інтелектуальні датчики використовуються для моніторингу різних промислових процесів, збору даних, проведення вимірювань і надсилання

даних на централізовані хмарні обчислювальні платформи, де інформація збирається та аналізується на закономірності. Ці дані можуть контролювати ключові особи, які приймають рішення, у будь-який час. Існує 4 основні типи інтелектуальних датчиків, які використовуються і забезпечують можливість впровадження Industry 4.0 на виробництвах.

Датчики рівня використовуються для вимірювання в режимі реального часу контейнерів, бункерів і резервуарів, передачі інформації в режимі реального часу в системи управління запасами та системи контролю процесів. Вони використовуються в усьому: від утилізації відходів до зрошення, вимірювання дизельного палива тощо.

Датчики температури є одним з найпоширеніших датчиків, що використовуються в промислових умовах. Часто використання подібних інтелектуальних датчиків потрібне для виявлення та запобігання перегріву, також для своєчасного інформування персоналу про необхідність технічного обслуговування або вимкнення.

Датчики тиску використовуються для моніторингу трубопроводів і сповіщення централізованої обчислювальної системи про витоки або порушення, які сповіщають керівників про необхідність технічного обслуговування та ремонту.

Інфрачервоні інтелектуальні датчики однаково багатоцільові та використовуються в дуже різних галузях. Вони використовуються в медицині для відстеження біологічних функцій, таких як кровотік під час операції, вони використовуються в архітектурі, інженерії та будівництві для моніторингу витоків тепла в будівлях і промислових об’єктах.

В залежності від задачі підприємства до розроблюваної системи можливо підключити будь який з представлених типів датчиків. В рамках дослідження розроблюваної системи мною буде використовуватися датчик температур з додаванням Wi-Fi модуля 2,4 ГГц ESP-01s ESP8266 та Arduino Mega 2560 у якості контролера для інтелектуального датчики.

TMP35, TMP36 і TMP37 – низьковольтні прецизійні стоградусні датчики

температури (рис 2.8). Вони забезпечують вихідну напругу, лінійно пропорційну температурі за шкалою Цельсія. TMP35/TMP36/TMP37 не потребує будь-якого зовнішнього калібрування для забезпечення типової точності ± 1 °C при + 25 °C та ± 2 °C у діапазоні температур від – 40 °C до + 125 °C. Низький вихідний імпеданс TMP35/TMP36/TMP37, лінійний вихід та точне калібрування спрощують взаємодію зі схемою контролю температури та аналого-цифровими перетворювачами. Всі три пристрої призначені для роботи з однополярним живленням від 2,7 В до 5,5 В максимум. Струм живлення значно нижче 50 мкА, що забезпечує дуже низький самонагрів, менше 0,1 °C в нерухомому повітрі. Крім того, передбачено функцію відключення, що дозволяє скоротити струм живлення до рівня менше 0,5 мкА. TMP35 функціонально сумісний з LM35/LM45 та забезпечує вихідну напругу 250 мВ при 25 °C. TMP35 зчитує температуру від 10 °C до 125 °C. TMP36 працює в діапазоні від - 40 °C до + 125 °C.

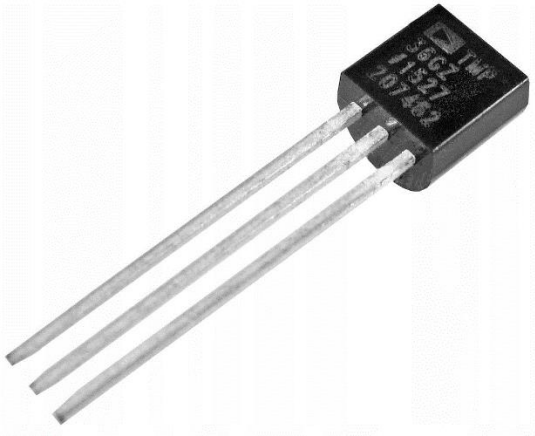


Рисунок 2.8 – Зовнішній вигляд температурного датчика TMP36gz

ESP8266 ESP-01s це модуль на основі 32-біт низькоспоживчого мікроконтролера з Wi-Fi модулем, інтегрованим TCP/IP стеком та управлінням AT-командами. Для програм передбачено 4 Мб SPI-flash. Модуль із високим ступенем інтеграції. Він споживає дуже мало енергії, забезпечуючи бездротове підключення до Інтернету для пристроїв, що вбудовуються. Він

може бути легко інтегрований у пристрої з обмеженими ресурсами через його малий форм-фактор всього 24,5 мм × 14 мм. На рисунку 2.9 зображена зовнішній вигляд модуля.

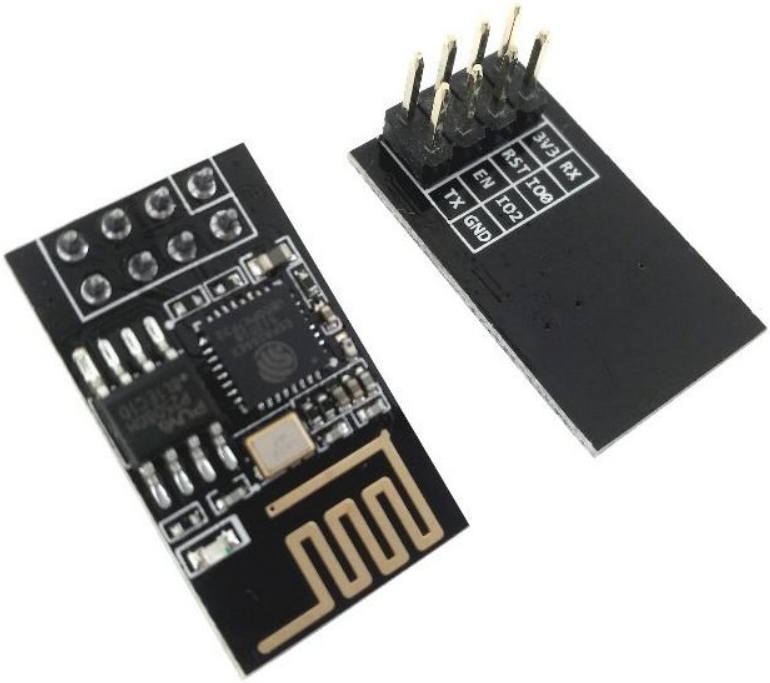


Рисунок 2.9 – Зовнішній вигляд Wi-Fi модуля ESP-01s ESP8266

Повний технічний опис модуля:

– підтримка Wi-Fi протоколів 802.11;

– Wi-Fi Direct (P2P), soft-AP;

– вбудований стек TCP/IP;

– вбудований TR перемикач, LNA, підсилювач потужності та відповідність мережі;

– вбудований PLL, регулятори, та система управління живленням;

– вихідна потужність +19,5 дБм у режимі 802.11b;

– SDIO 2.0, SPI, UART;

– STBC, 1×1 MIMO, 2×1 MIMO;

– пробудження та відправлення пакетів до 22 мс;

– споживання в режимі Standby до 1,0 мВт;

– розміри: 24,5 мм × 14 мм.

Варіанти застосування модуля:

– інтеграція до системи розумного будинку;

– домашня метеостанція з переглядом показів онлайн;

– облік показань лічильників води, електролічильників та перегляд показань онлайн;

– керована по Wi-Fi розетка, люстра або інші електроприлади;

– вбудовування у вимикачі з локальним та віддаленим керуванням.

Arduino Mega побудована на мікроконтролері ATmega2560. Плата має 54 цифрових входів/виходів, 14 з яких можуть використовуватися як виходи ШІМ, 16 аналогових входів, 4 послідовних порту UART, кварцовий генератор 16 МГц, USB конектор, роз'єм живлення, роз'єм ICSP і кнопка перезавантаження. Для роботи необхідно підключити платформу до комп'ютера за допомогою кабелю USB або подати живлення за допомогою адаптера AC/DC або акумуляторною батареєю. Arduino Mega 2560 сумісна з усіма платами розширення, розробленими для платформ Uno або Duemilanove [30]. На рисунку 2.10 зображена зовнішній вигляд плати, крім того у таблицю 2.4 було вказано основні характеристики плати.

Рисунок 2.10 – Загальний вигляд Arduino Mega 2560 [30]

Таблиця 2.4 – Характеристика Arduino Mega 2560 [30]

Робоча напруга	5 В
Вхідна напруга (рекомендована)	від 7 В до 12 В
Вхідна напруга (гранична)	від 6 В до 20 В
Цифрові Входи/Виходи	54, 14 з яких можуть працювати також як виходи ШІМ
Аналогові входи	16
Постійний струм через вхід/вихід	40 мА
Постійний струм для виведення	50 мА
Флеш-пам'ять	256 кб

2.4 Висновки до другого розділу

В ході дослідження було розроблена структурну схему системи візуалізацій та моніторингу на базі якої розроблено інформаційної моделі системи. На якій зображено взаємозв’язок між всіма складовими системи та напрям руху даних всередині системи. Використовуючи отримані дані була створена архітектура для розроблюваної системи з вказанням всіх основних компонентів. Крім того для розробки системи та реалізації її апаратної частини було проведено аналіз та вибір апаратних модулів. В результаті якого було обрано Raspberry Pi 4 Model B на базі якого буде створюватися розроблювана система. Для тестування роботи розроблюваної системи буде використовуватися датчик температури TMP36gz, який з допомогою додаткових модулей буде передавати дані до системи.

3 РОЗРОБКА АВТОМАТИЗОВАНОЇ СИСТЕМИ МОНІТОРИНГУ ТА ВІЗУАЛІЗАЦІЙ

3.1 Вибір середовища розробки та СУБД

Середовище розробки програми було обрано PhpStorm — це інтегроване середовище розробки (IDE) від компанії JetBrains для мови програмування PHP та інших веб-технологій, таких як HTML5, CSS або JavaScript. Функції включають рефакторинг, інтелектуальне підсвічування коду та синтаксису, підтримку PHPUnit, інструменти контролю версій і кілька способів автоматичного створення коду. PhpStorm базується на IntelliJ IDEA від JetBrains, але представляє специфічну версію PHP [31].

Діапазон функцій можна розширити за допомогою плагінів , які частково розроблені JetBrains, а частково спільнотою .

Функції включають, серед іншого, швидкий запуск, швидке автозавершення, інтелектуальне підсвічування синтаксису, інтеграцію VCS , автоматичне розгортання та функції оптимізації коду.

Arduino IDE — інтегроване середовище розробки для Windows, MacOS та Linux, розроблене на Ci та C++, призначене для створення та завантаження програм на Arduino-сумісні плати, а також на плати інших виробників.

Вихідний код середовища випущений під загальнодоступною ліцензією GNU версії 2. Підтримує мови Ci і C++ з використанням спеціальних правил структурування коду. Arduino IDE надає бібліотеку програмного забезпечення з проєкту Wiring, яка надає безліч загальних процедур введення та виведення. Для написаного користувачем коду потрібні лише дві базові функції для запуску ескізу та основного циклу програми, які скомпільовані та пов'язані із заглушкою програми main() у виконувану циклічну програму з ланцюжком інструментів GNU, також включеною до

дистрибутиву IDE.

У якості СУБД в даній роботі буде використовуватися MySQL, яка представляє собою одна з найпопулярніших баз даних.

MySQL не відноситься до профільних систем корпоративного програмного забезпечення, але фактично є стандартом для веб-серверів, які працюють під управлінням операційної системи Linux. MySQL – це безкоштовний пакет програм, розробку і підтримку якого здійснює корпорація Oracle. Незважаючи на довге знаходження на ринку СУБД MySQL регулярно отримує оновлення у яких, розширюється доступний функціонал і покращується безпека СУБД. Окрім безкоштовної версії існує спеціальна платна версії, призначені для комерційного використання. Основним напрямком безкоштовної версії є швидкість і надійність, що є дуже важливим для роботи на підприємстві з провадженою концепцією Industry 4.0. Гнучкість СУБД MySQL забезпечується підтримкою великої кількості типів таблиць: користувачі можуть вибрати як таблиці типу MyISAM, що підтримують повнотекстовий пошук, так і таблиці InnoDB, що підтримують транзакції на рівні окремих записів. Система має достатній рівень надійності і підходить для систем, які потрібують надійний, але безкоштовний інструмент управління базами даних [32].

3.2 Розробка загального алгоритму роботи системи

Для подальшої побудови системи було розроблено загальний алгоритм системи. Розроблений алгоритм має загальний характер і охоплює всі етапи роботи системи. Узагальнений алгоритм роботи макету системи моніторингу та візуалізації даних для кіберфізичних виробничих систем представлено на рис. 3.1.

Рисунок 3.1 – Узагальнений алгоритм роботи макету системи моніторингу та візуалізації даних для кіберфізичних виробничих систем

У відповідності до узагальнений алгоритм роботи макету першим етапом алгоритму роботи буле підключення датчиків обраних у другому розділі. Для виконання цього етапу дані про датчики будуть занесені до бази даних розміщеній на контролері. Після чого використовуючи ці дані буде проводитися підключення датчиків через бездротове з’єднання.

Другий етап представляє собою перевірка підключення між контролером та датчиком шляхом відправки декількох сигналів до датчиків та відповідей на них. Крім того перевірка стабільності інтернет підключення.

API адреса так само, як і дані про датчики зберігаються у локальній базі даних. На четвертому та п’ятому етапах ці адреса зчитуються з базі даних та перевіряються їх активність, тобто наявність доступу до збережених адресів. У разі успішного проходження всіх попередніх етапів починається роботи основної частини алгоритму роботи системи.

Процес збору даних з датчиків поділений на два етапі. Перший, це сам процес отримання необроблених даних з датчиків, а до другого етапу відноситься первинна обробка отриманих даних та збереження їх до локальної бази даних. Це необхідно, щоб у разі тимчасової відсутності з’єднання з віддаленими серверами дані не були втрачені.

Наступним етапом нові дані з бази даних зберігаються у форматі JSON (JavaScript Object Notation). Цей формат представляє собою текстовий формат обміну даними між комп'ютерами. Формат дає змогу описувати об'єкти та інші структури даних. Використовується переважно для передачі структурованої інформації через мережу [33]. Після чого отриманий набір даних відправляється за вказаними у базі даних API адресами.

На етапі отримання даних сервером, пакет даних отриманий з контролера розшифровується та зберігається у базу даних сервера з точним часом, коли ці дані були зафіксовані датчиками.

Останнім етапом є зчитування даних з бази даних та відображення їх на пристрої користувача за допомогою веб-сайту, основними інструментами для чого буде фреймворк для PHP Laravel, який буде відповідати за обробку та зберігання даних, та фреймворк для JavaScript Vue.js, який буде відповідати за візуальну частину веб-сайту та можливість динамічного відображення даних без необхідності перезавантаження сторінки.

3.3 Реалізація функцій системи

3.3.1 Функція отримання даних з датчика та відправки до системи

На першому етапі було створено програму для підключення датчиків для отримання даних з них та модуль для бездротової передачі даних. У даній програмі для сумісної роботи всіх модулей було обрано метод передачі даних

через UART. Для цього були підключені відповідні піни RX та TX, за задано швидкість передачі даних. Після чого проведено підключене Wi-Fi модуля до Wi-Fi. За допомогою створеної функції getTemp() отримуються дані з датчика та перетворюємо їх у цілочислене значення температури. Після чого створюється масив значень у форматі JSON. Наступними командами відбувається підключення модуля до віддаленого серверу. У разі успішного виконання підключення відбувається побудова HTTP POST запиту за допомогою якого дані передаються до наступного модуля системи, а саме модуля візуалізації. Після чого відбувається закриття з'єднання та програма повертається на етап отримання даних з датчика (рис 3.2).

```
10
11  void setup() {
12    Serial.begin(115200);
13    Serial1.begin(115200);
14    sendCommand("AT", 5, "OK");
15    sendCommand("AT+CWMODE=1", 5, "OK");
16    sendCommand("AT+CWJAP=\""+ AP +"\",\""+ PASS +"\"", 20, "OK");
17  }
18
19  void loop() {
20    if (Serial.available()) { Serial1.write(Serial.read()); }
21    getTemp();
22    valSensor = temp;
23    String jSON = "{\"sensor_id\":\"1\",\"value\":\""+ String(temp) +"\"}";
24    unsigned int l = String(temp).length();
25    unsigned int i = 200- (190 + l);
26    sendCommand("AT+CIPMUX=0",5,"OK");
27    sendCommand("AT+CIPSTART=\"TCP\",\""+ HOST +"\","+ PORT,16,"OK");
28    sendCommand("AT+CIPSEND=200", 4,">");
29    Serial1.println("POST /api/sensors-data-store HTTP/1.1");
30    Serial1.println("Host: "+ HOST +":"+ PORT +"");
31    Serial1.println("User-Agent: insomnia/2022.6.0");
32    Serial1.println("Content-Type: application/json");
33    Serial1.println("Accept: */*");
34    Serial1.println("Content-Length: 35");
35    Serial1.println("");
36    Serial1.println(jSON);
37    while(i > 0) {
38      Serial1.println(""); i--;
39    }
40    delay(4000);
41    countTrueCommand++;
42    sendCommand("AT+CIPCLOSE=0",5,"OK");
43  }
44
45  void getTemp(){
46    int reading = analogRead(A5);
47    float voltage = reading * 5.0;
48    voltage /= 1024.0;
49    Serial.print(voltage); Serial.println(" volts");
50    float temperatureC = (voltage - 0.5) * 100 ;
51    Serial.println("Temperature = " + String(temperatureC));
52    Serial.println("Volts = " + String(voltage));
53    temp =  temperatureC + 0.5;
54  }
```

Рисунок 3.2 – Код отримання даних з датчика та відправки до системи

3.3.2 Створення API точок

На цьому етапі були створені всі головні точки доступу до системи. Вони поділяються на точки для роботи з процесами, сенсорами та окремо даними з датчиків. В ході роботи були створені точки для створення, редагування, показу та видалення даних з бази даних за допомогою них. На рисунку 3.3 зображено код оголошення всіх точок.

```php
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Route;
use App\Http\Controllers\Api\v1\ProcessesController;
use App\Http\Controllers\Api\v1\SensorsController;
use App\Http\Controllers\Api\v1\SensorsDataStoreController;

/*
|--------------------------------------------------------------------------
| API Routes
|--------------------------------------------------------------------------
|
| Here is where you can register API routes for your application. These
| routes are loaded by the RouteServiceProvider within a group which
| is assigned the "api" middleware group. Enjoy building your API!
|
*/

Route::get('process/{id}/{range}/range', [ProcessesController::class, 'showSensorsRangeByProcessId'])->name('process.showSensorsRangeByProcessId');
Route::get('process/{id}/sensors', [ProcessesController::class, 'getSensorsById'])->name('process.getSensorsById');
Route::resource('process', ProcessesController::class);
Route::get('sensor/{id}/{range}/range', [SensorsController::class, 'showRangeById'])->name('sensors.showRangeById');
Route::resource('sensor', SensorsController::class);
Route::get('sensors-data-store/{range}/range', [SensorsDataStoreController::class, 'showRange'])->name('sensors-data-store.showRange');
Route::resource('sensors-data-store', SensorsDataStoreController::class);
```

Рисунок 3.3 – Код оголошення API точок системи

3.3.3 Функція збереження даних

Функція store необхідна для отримання даних відправлених через API. У цій функції відбувається валідації отриманих даних, в результаті якої, у разі не проходження, відправляється інформація про помилку та помилковий статус, а у разі успішного проходження валідації відбувається збереження даних та відправлення відповіді з позитивним статусом. На рисунку 3.4 зображено код функції store.

```php
/**
 * Store a newly created resource in storage.
 *
 * @param  \Illuminate\Http\Request  $request
 * @return \Illuminate\Http\Response
 */
public function store(Request $request)
{
    $validator = Validator::make(
        $request->all(), [
            'sensor_id' => ['required'],
            'value' => ['required'],
        ]
    );

    if ($validator->fails()){
        return [
            'status' => false,
            'errors' => $validator->messages()
        ];
    }

    $sensorsDataStore = SensorsDataStore::create([
        'sensor_id' => $request->sensor_id,
        'value' => $request->value,
    ]);

    return [
        'status' => true,
        'process' => $sensorsDataStore
    ];
}
```

Рисунок 3.4 – Код функції store

3.3.4 Функція отримання даних з БД

Логіка цієї функції включає в себе запит на отримання даних з бази даних згідно з параметрами вказаними у API точці. До цих параметрів входить унікальний ідентифікатор процесу на якому ці дані отримані, а також кількість значень які необхідно отримати. Сам запит складається з декількох частин, а саме отримання всіх сенсорів, що використовуються у процесі, після чого згідно їх унікального ідентифікатора отримати необхідну кількість значень для кожного датчика. На рисунку 3.5 зображено код функції showSensorsRangeByProcessId.

```php
/**
 * Display the range of resource by sensor.
 *
 * @param  int  $id
 * @param  int  $range
 * @return \Illuminate\Http\Response
 */
public function showSensorsRangeByProcessId($id, $range)
{
    $sensors = Process::findOrFail($id)->processSensors()->get();
    $dataList = collect([]);
    foreach ($sensors as $sensor) {
        $dataList = $dataList->merge( [
            [
                $sensor->name, Sensor::findOrFail($sensor->id)->sensorData()->latest()->take($range)->get()
            ]


        ]);
    }
    return $dataList;
    //return Process::findOrFail($id)->sensorsData()->latest()->take($range)->get();
}
```

Рисунок 3.5 – Код для отримання даних датчиків по одному процесу

3.3.5 Функція візуалізації даних

Ця функція представляє собою скрипт для отримання і виводу даних та є основною для системи візуалізації, складається з декількох простіших по своєму призначенну методів.

Метод loadChartData виконує фунцію отримання даних з системи через запит для спецільного api адресу, у якому необхідно вказати кількість значень що необхідно отримати та id процесу, даны з якого потрібно візуалізувати.

Метод createChart перетворую отриманны даны у лінійну діаграму, яка відображається у системы.

На рисунку 3.6 зображено код всієї функції.

```
<script>
    import axios from 'axios';

    export default {
        name: 'Index',
        data() {
            return {
                range: 10,
                process_id: 1,
                timer: '',
                chartData: []
            }
        },
        created () {
            this.fetchEventsList();
            this.timer = setInterval(this.fetchEventsList, timeout: 5000);
        },
        methods: {
            fetchEventsList () {
                console.log('load');
                this.loadChartData();
                this.createChart();
            },
            loadChartData () {
                axios.get('/api/process/'+ this.process_id +'/' + this.range + '/range')
                .then(res => {
                    this.chartData = res.data;
                })
            },
            createChart () {
                var chartsData = [];
                for (let k = 0; k < this.chartData.length; k++){
                    let chartLine = [];
                    for (let i = 0; i < this.chartData[k][1].length; i++){
                        chartLine.push(this.chartData[k][1][i].value)
                    }
                    chartLine.reverse()
                    chartLine.unshift(this.chartData[k][0])
                    chartsData.push(chartLine)
                }

                bb.generate( config: {
                    data: {
                        columns: chartsData,
                        type: "line",
                    },
                    bindto: "#lineChart"
                });
            },
        }
    }
</script>
```

Рисунок 3.6 – Код функції візуалізації даних

3.4 Розробка логічної та фізичної моделі бази даних

Логічна модель даних, або логічна схема — представляє собою модель даних для предметної області, створюється незалежно від конкретного продукту керування базами даних або технології зберігання, але в термінах структур даних, таких як реляційні таблиці та колонки, об'єктно-орієнтовані класи чи теги XML. Вона є протилежністю концептуальній моделі даних, яка описує семантику організації без посилання на технологію.

В ході роботи над системою було розроблено комплекс баз даних, яка складається з двох баз даних. Одна з яких буде встановлена на Raspberry Pi 4, як система управління та друга яка буде встановлена на віддаленому сервері та буде взаємодіяти з системою візуалізації. Взаємодія між частинами комплексу буде відбуватися через REST API, з конвертацією даних у JSON формат для передачі їх між частинами системи На рисунку 3.7 зображено комплексну логічну модель баз даних розроблюваної системи.

Рисунок 3.7 – Комплексна логічна модель баз даних розроблюваної системи

Логічну модель бази даних для Raspberry Pi 4 можна побачити на

рисунку 3.8. До даної моделі входять декілька таблиць, які відповідають за інформацію про датчики, процеси на підприємстві на яких потрібні датчики, інформація про необхідні API адреса, на які будуть відправлятися дані для візуалізації, та загальна таблиця з даними з датчиків, крім того к кожному запису з датчика додається дані про процес, на якому дані було отримано, та датчик, який отримав ці данні.

Рисунок 3.8 – Логічна модель бази даних для Raspberry Pi 4

Логічну модель бази даних для на віддаленого серверу можна побачити на рисунку 3.9 . Дана модель є похідною від моделі для системи управління, але в ній є деякі відмінності. Для коректної візуалізації інформації було додано таблицю з даними о допустимих значення для того, чи іншого датчика на конкретному процесі. Ці данні дозволять розроблювати більш гнучку системи візуалізацію з можливістю використовувати ті самі датчики на різних процесах, навіть якщо допустимі значення в них сильно відрізняються. Крім того ця модель не потребу таблиці зі списком API адресів.

Рисунок 3.9 – Логічна модель бази даних для віддаленого серверу

Фізична модель даних — представляє собою спосіб подання дизайну даних, як вже реалізованого чи призначеного для реалізації у системі керування базами даних. У більшості життєвих циклів проєктів подібні моделі походять від логічних моделей даних, але в деяких випадках може бути зворотно розроблена з даної реалізації бази даних. Завершена фізична модель даних включатиме всі артефакти бази даних, необхідні для створення відношень між таблицями чи для досягнення мети продуктивності, як-от індексів, визначень обмежень, зв'язаних і секціонованих таблиць або кластерів.

На рисунку 3.10 зображено комплексну фізична модель баз даних розроблюваної системи.

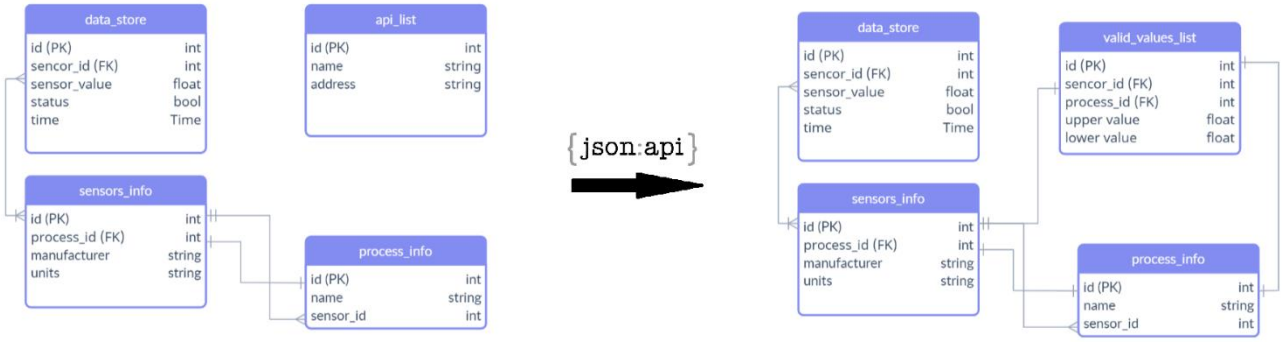


Рисунок 3.10 – Комплексна фізична модель баз даних для розроблюваної системи

Фізична модель бази даних для Raspberry Pi 4 можна побачити на рисунку 3.11.

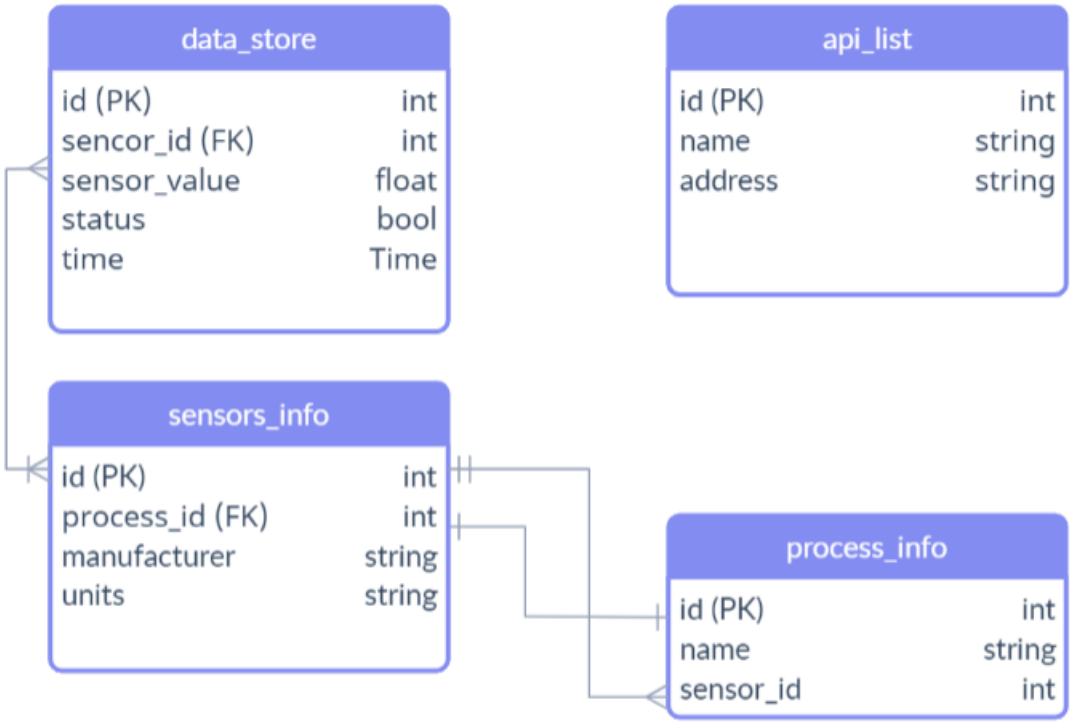


Рисунок 3.11 – Фізична модель бази даних для Raspberry Pi 4

Фізична модель бази даних для віддаленого серверу можна побачити на рисунку 3.12 .

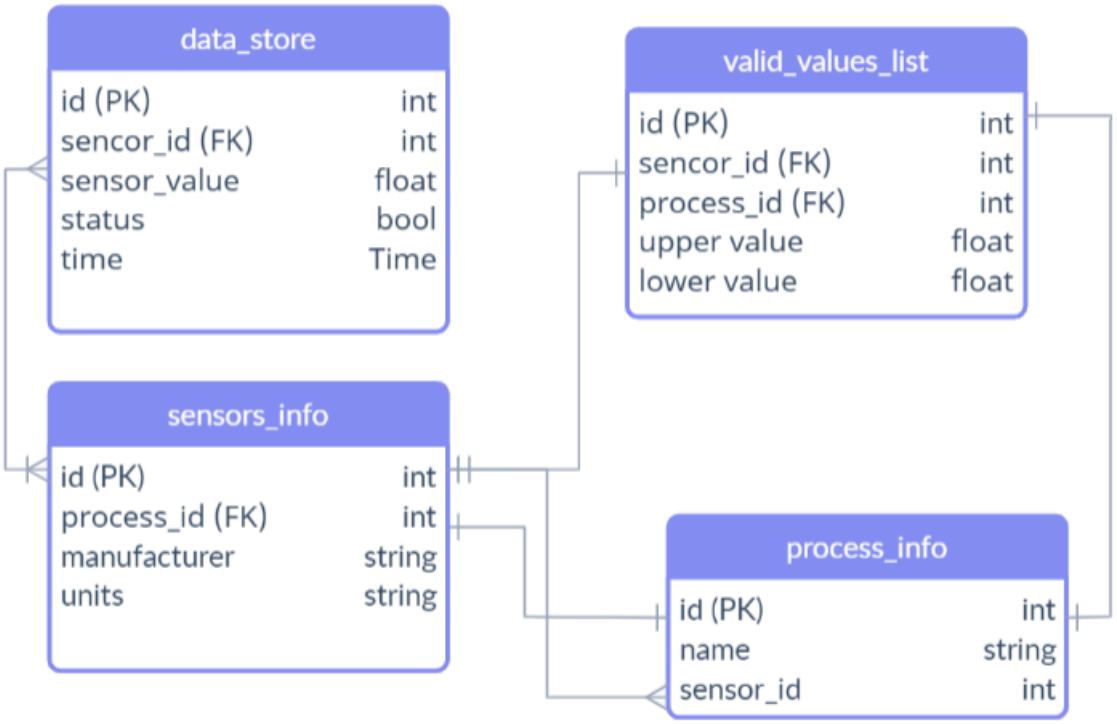


Рисунок 3.12 – Фізична модель бази даних для віддаленого серверу

На рисунку 3.13. зображено всі класи для створення таблиць бази даних у системі. Всі класи складаються з двох методів, up, що відповідає за створення чи модифікацію таблиці та down, що виконується у випадку видалення таблиці. Таблиця створюється за допомогою статичного метода create, у параметри якого вписується назва таблиці та безіменну функцію у якій виконується ініціювання всіх полів таблиць та зв'язки між ними. Метод id вказує на унікальний ідентифікатор запису, string та integer створюють поля з відповідним типом даних, а unsignedBigInteger вказує на поле яке буде використовуватися для зовнішнього ключа. Для коректного підключення зовнішнього ключа також потрібно додати функцію foreign у якій вказуються взаємозв’язок між таблицями та конкретними полями.

```php
class CreateProcessesTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('processes', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('processes');
    }
}
```

```php
class CreateSensorsDataStoresTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('sensors_data_stores', function (Blueprint $table) {
            $table->id();
            $table->unsignedBigInteger('sensor_id');
            $table->integer('value');
            $table->timestamps();

            $table->foreign('sensor_id')->references('id')->on('sensors');
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('sensors_data_stores');
    }
}
```

```php
class CreateSensorsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('sensors', function (Blueprint $table) {
            $table->id();
            $table->unsignedBigInteger('process_id');
            $table->string('name');
            $table->string('unit');
            $table->timestamps();

            $table->foreign('process_id')->references('id')->on('processes');
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('sensors');
    }
}
```

```php
class CreateValidValuesListTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('valid_values_list', function (Blueprint $table) {
            $table->id();
            $table->unsignedBigInteger('process_id');
            $table->unsignedBigInteger('sensor_id');
            $table->integer('upper_value');
            $table->integer('lower_value');
            $table->timestamps();

            $table->foreign('process_id')->references('id')->on('processes');
            $table->foreign('sensor_id')->references('id')->on('sensors');
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('valid_values_list');
    }
}
```

Рисунок 3.13 – Класи міграцій таблиць процесів, датчиків, даних з датчиків та допустимих значень

3.5 Реалізація візуалізацій інформацій у SCADA/HMI

Для даної системи буде реалізовано клієнт-серверну архітектуру на базі фреймворків Laravel та Vue JS з можливістю користувача звертатися до тонкого клієнта для отримання даних з датчиків у реальному часі.

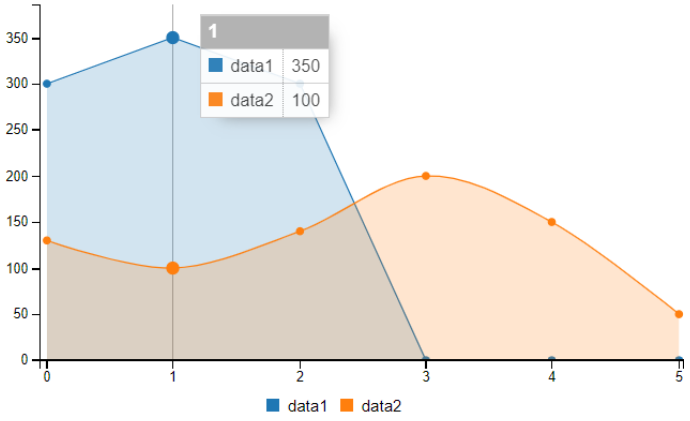
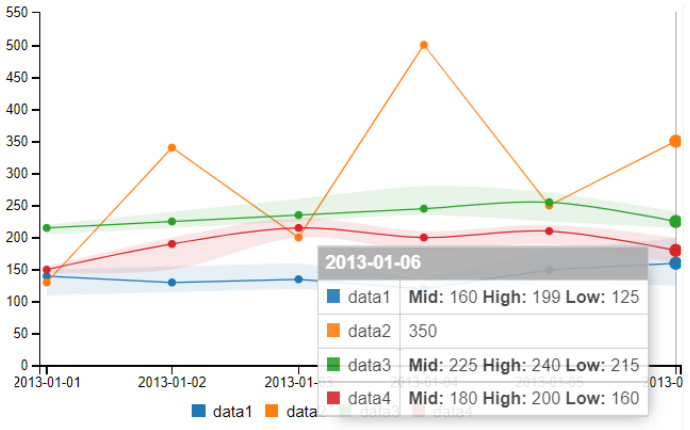
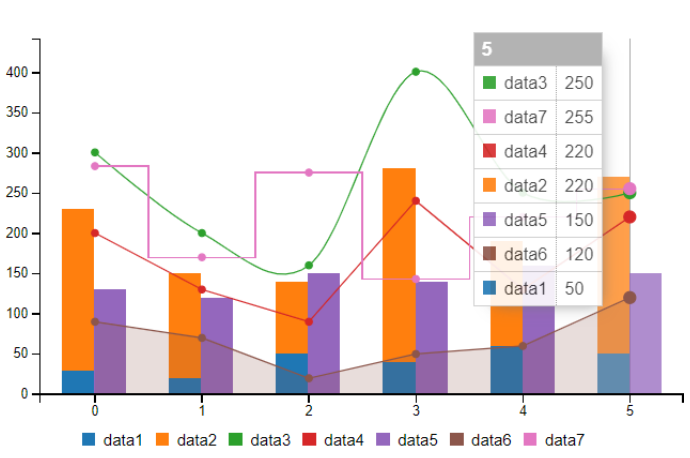
“Клієнт — сервер” (client–server) — мережева архітектура, основною ідеєю якої є розподілення завдань чи мережевого навантаження між постачальниками послуг, званими серверами, та замовниками послуг,

званими клієнтами. У більшості випадків програми розташовані на різних обчислювальних машинах і взаємодіють між собою через обчислювальну мережу за допомогою мережевих протоколів, але можуть бути розташовані також і на одній машині. Програми-сервери очікують від клієнтських програм запити та надають їм свої ресурси у вигляді даних або у вигляді сервісних функцій. Однак програма-сервер може виконувати запити від багатьох програм-клієнтів, її розміщують на спеціально виділеній обчислювальній машині, налаштованій особливим чином, як правило, спільно з іншими програмами-серверами, тому продуктивність цієї машини повинна бути високою. Через особливу роль такої машини в мережі, специфіки її обладнання та програмного забезпечення її також називають сервером, а машини, що виконують клієнтські програми, відповідно, клієнтами [34].

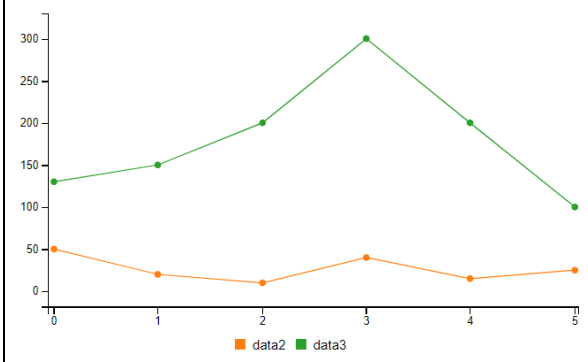
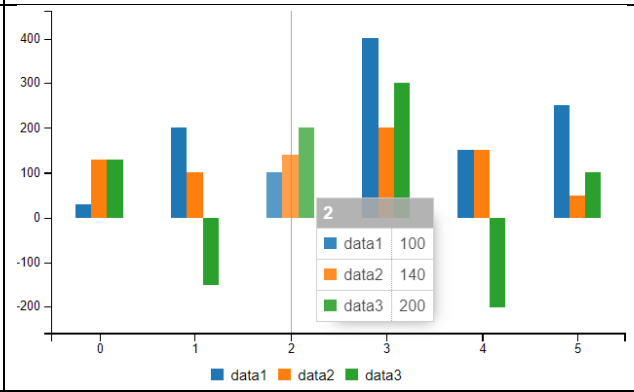
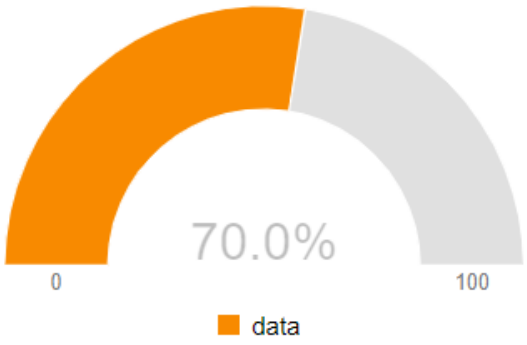
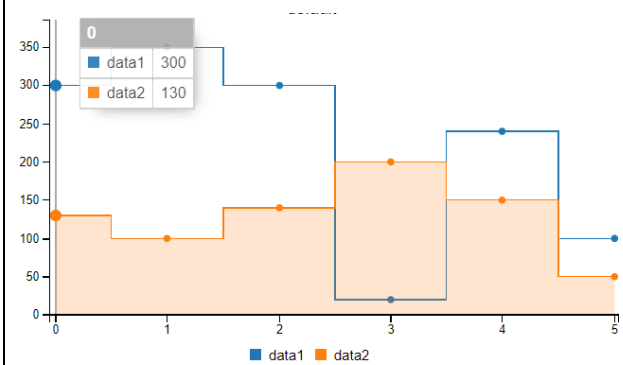
Тонкий клієнт – це у більшості випадків низькопродуктивний комп’ютер, оптимізований для встановлення віддаленого з’єднання з обчислювальним середовищем на основі сервера. Також може називатися мережевими комп'ютерами або нульовими клієнтами. Сервер виконує більшу частину роботи, яка може включати запуск програм , виконання обчислень і зберігання даних [35].

Візуалізацій інформацій у даній системі буде виконуватися за допомогою бібліотеки “billboard.js” на базі фреймворку Vue JS. Ця бібліотека була обрана бо вона дає багато інструментів, які забезпечують адаптивний та інтерактивний вид готової системи. У таблиці 3.1 зображені основі види візуалізації інформації з бібліотеки, які можуть бути використані у розроблюваній системі. Окрім лінійних діаграм є можливість використання секторних та кільцевих діаграми, гістограм, діаграм з областями, точкових та бульбашкових діаграм, поверхневих діаграм, пелюсткових діаграми та інших. Подібний широкий набір інструментів візуалізації дозволяє настроїти системи під потреби більшості підприємств.

Таблиця 3.1 – Основі види візуалізації інформації з бібліотеки billboard.js [36]

Опис діаграм	Зображення діаграм
1	2
Діаграми з областями, використовуються для зображення зміни кількісних значень в певному інтервалі або за певний період часу. Найчастіше ці діаграми використовують для демонстрації тенденцій, а не конкретних значень.	
Діаграми плоскостей – це лінійні діаграми, що відображають діапазон між нижчим і вищим значенням для кожної точки. Діаграми площі зазвичай використовуються для візуалізації діапазону, який змінюється з часом.	
Комбінована діаграма порівнює дані в кількох різних категоріях за певний період часу. Загалом існує два або три типи діаграм, об’єднаних разом, щоб показати зв’язок між точками даних. Як правило, гістограма та лінійна діаграма використовуються разом.	

Продовження таблиці 3.1

1	2
Лінійна діаграма відображає інформацію як серію точок даних, з'єднаних відрізками. Точки впорядковують за однією віссю. Лінійна діаграма часто використовується для візуалізації тенденції в даних через інтервали часу.	
Стовпчикова діаграма — це графік, який представляє згруповані дані, за допомогою прямокутних стовпців довжини яких пропорційні значенням, які вони представляють.	
Циферблатні діаграми, використовуються для відображення інформації як показання на циферблаті. Цей тип діаграми часто використовується у звітах на інформаційній панелі для відображення ключових індикаторів системи.	
Покрокова діаграма – це лінійна діаграма, яка не використовує найкоротшу відстань для з’єднання двох точок даних. Замість цього він використовує вертикальні та горизонтальні лінії для утворення ступінчастої прогресію. Вертикальні частини покрокової діаграми позначають зміни в даних та їх величину.	

3.6 Висновки до третього розділу

В ході дослідження було розроблена система моніторингу та візуалізації. Для цього було обрано середовища розробки та СУБД, а саме PhpStorm та MySQL. Для подальшої роботи було розроблені логічної та фізичної моделі бази даних на основі яких додані таблиці баз даних до основної бази даних системи. Розроблено загальний алгоритм роботи системи з детальним описанням всіх етапів роботи системи. Після чого реалізовано саму систему. Отриману системи було розбито на окремі функції, які були описані та проаналізовані. Останнім етапом було реалізація візуалізації інформації у SCADA/HMI.

4 ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА ПЕРЕВІРКА ПРАЦЕЗДАТНОСТІ СИСТЕМИ

4.1 Розробка макета та постановка завдань експерименту

Розроблюваний макет системи складається з двох основних модулей, а саме модуля моніторингу та візуалізації. Модуль моніторингу представляє собою інтелектуальний датчик на основі температурного датчика TMP36gz, Arduino Mega 2560 та Wi-Fi модулем ESP8266 ESP-13. Схема підключення представлена на рисунку 4.1.

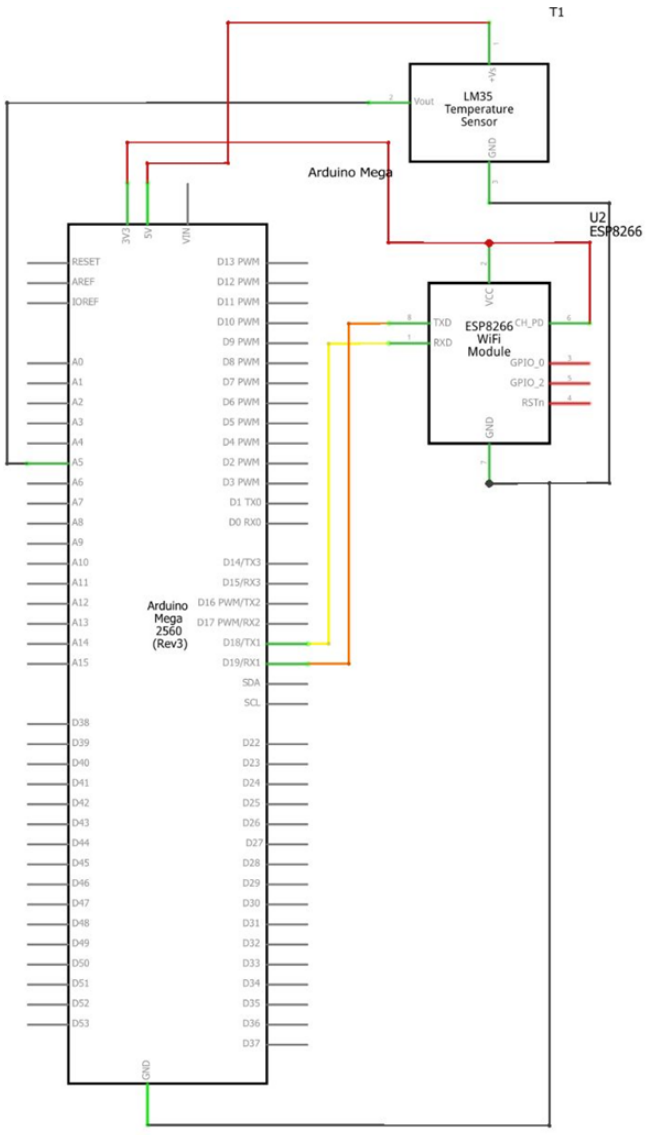


Рисунок 4.1 – Схема підключення модуль моніторингу

На рисунку 4.2 представлено схематичне підключення компонентів модуля.

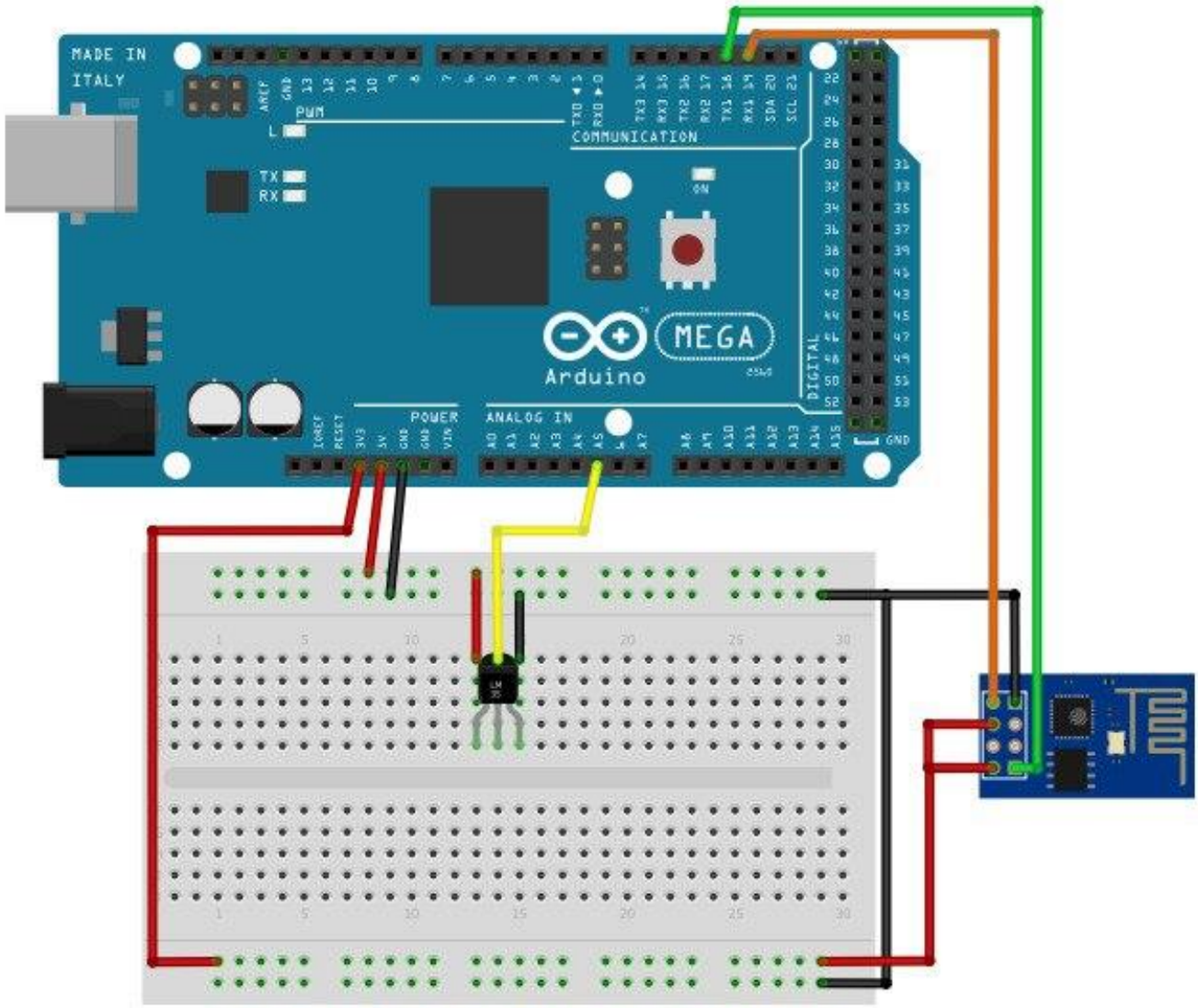


Рисунок 4.2 – Схематичне підключення компонентів модуля моніторингу

На рисунку 4.3 представлено зовнішній вигляд модуля.

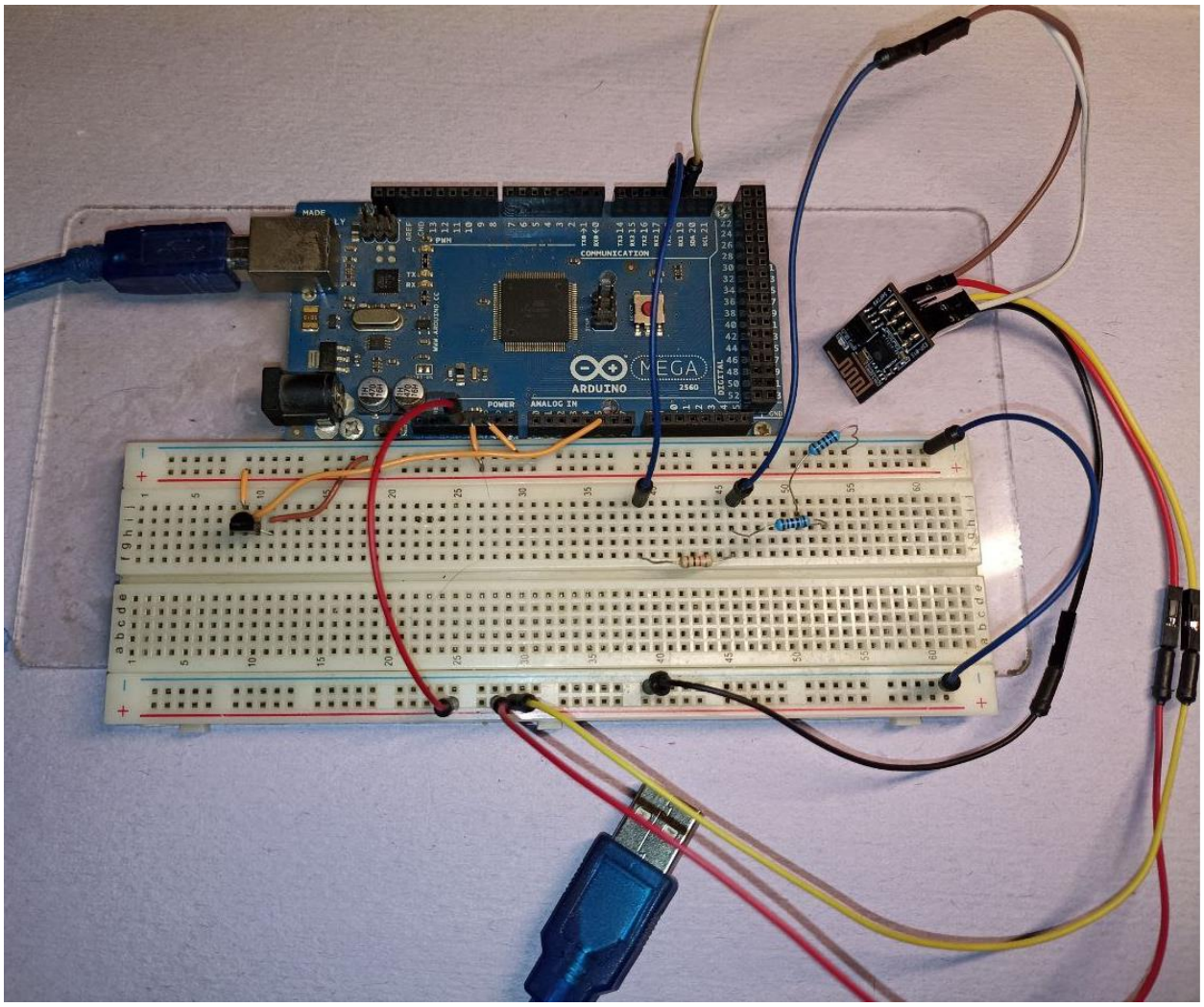


Рисунок 4.3 – Зовнішній вигляд модуля моніторингу

Модуль візуалізації представляє собою одноплатний комп'ютер Raspberry Pi 4 Model B на який було встановлено операційну систему Linux, а саме дистрибутив Debian, сервер Apache2, базу даний MySQL та мову програмування PHP.

Для перевірки коректності роботи модуля до нього також було підключено периферійні пристрої, а саме дисплей, клавіатура та миша.

Схематичний вид Raspberry Pi 4 Model B представлена на рисунку 4.4.

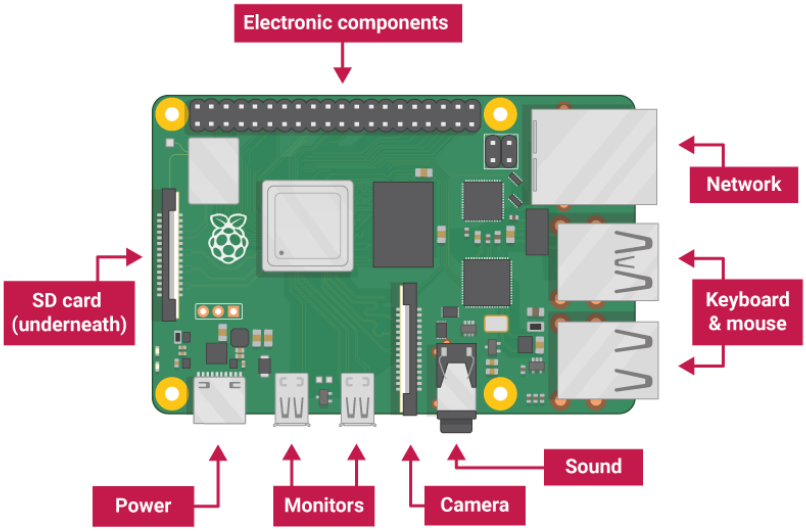


Рисунок 4.4 – Схематичний вигляд модуля візуалізації

На рисунку 4.5 представлено зовнішній вигляд модуля.

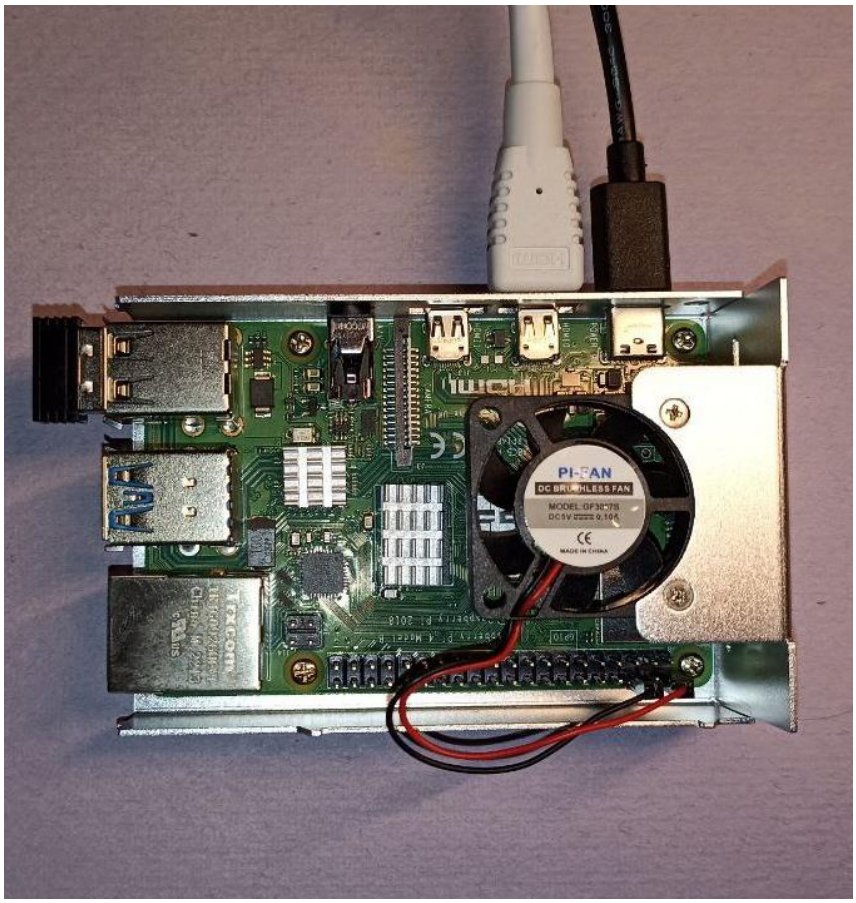


Рисунок 4.5 – Зовнішній вигляд модуля візуалізаці

Після збирання та підключення усіх елементів системи було проведено експеримент. Завданням експерименту є перевірка швидкості передачі інформації в часі в залежності від відстані от датчика до сервера. Ця задача обумовлена модулюванням ситуації на підприємстві, коли датчикі знаходяться на відстані від контролера та потрібно передавати дані до нього з мінімальною затримкою. Тобто у експерименті було перевірено швидкість реакції HMI на надходження інформації після спрацювання датчика. У якості інструменту візуалізації даних було використано лінійну діаграму, яка демонструє зміну значень датчика у реальному часі. На рисунку 4.6 представлено вигляд розробленої HMI.

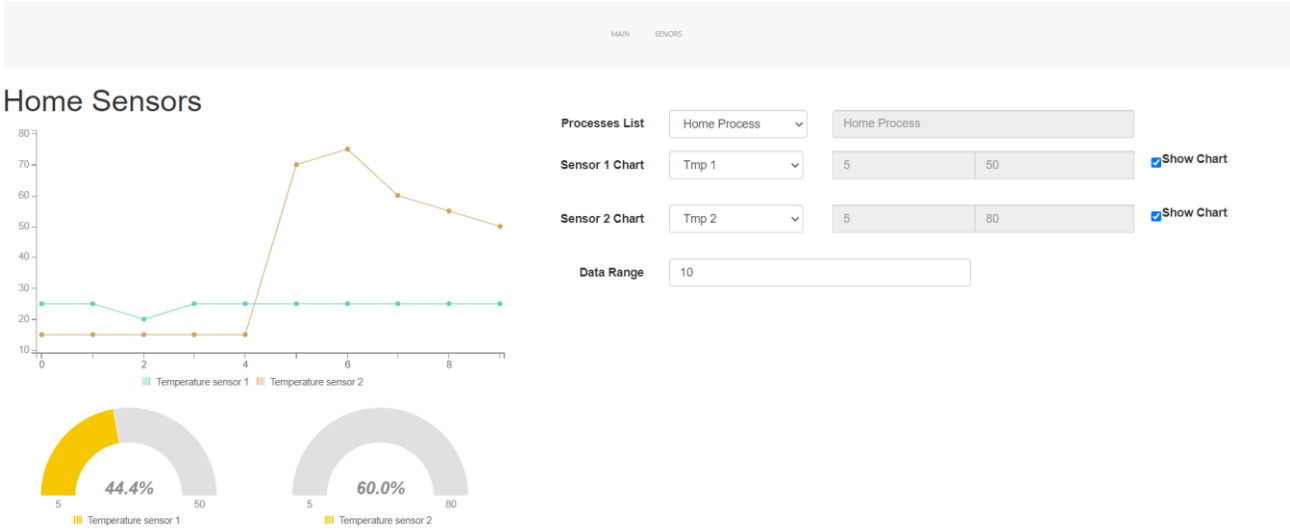


Рисунок 4.6 – Вигляд розробленої HMI

4.2 Проведення дослідження та аналіз отриманих результатів

Для виконання завдання було обрано 4 дистанції, а саме 0 м, 5 м, 10 м та 15 м. Першим етапом було дослідження втрат сигналу в залежності від відстані. Ці дані були отримані за допомогою спеціального програмного забезпечення, для вимірювання потужність сигналу пристрою. Отримані дані були занесені у таблицю 4.1 та представлені на рисунку 4.7 у вигляді графіку.

Діапазон потужності сигналу до - 73 дБм вважається, що це майже не впливає на якість сигналу та пропускну здатність даних, від - 75 дБм до - 93 дБм, це непоганий сигнал, від - 95 дБм і нижче поганий сигнал.

Таблиця 4.1 – Значення потужності сигналу в залежності від відстані та перешкод

Дальність, м	Кількість перешкод		
	Без перешкод	Одна стіна	Дві стіни
0	- 17 дБм	- 22 дБм	- 35 дБм
5	- 48 дБм	- 55 дБм	- 68 дБм
10	- 58 дБм	- 72 дБм	- 87 дБм
15	- 70 дБм	- 99 дБм	- 111 дБм

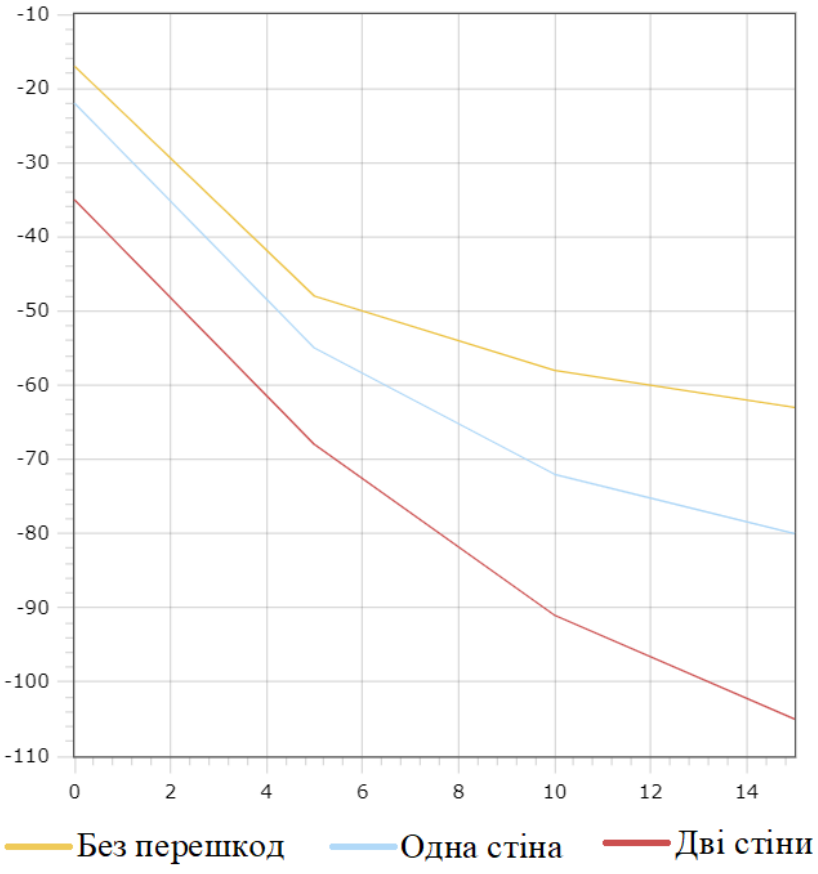


Рисунок 4.7 – Графік залежності втрат сигналу до відстані та перешкод

З отриманих даних можна зробити висновок, що без перешкод втрати потужності не являються критичними і навіть на максимальній досліджуваній дальності сигнал є досить якісним. Також можна побачити, що наявність однієї перешкоди (стіни) впливає на сигнал і вже на 10 м втрати сигналу можуть впливати на процес передачі даних. Наявність 2 чи більше перешкод (стін) сильно впливає на потужність сигналу, що може сильно вплинути на якість та чіткість отриманих даних. Крім того слід зауважити, що матеріал з якого зроблені перешкоди також сильно впливає на результати експерименту. В залежності від щільності та товщиною матеріалу, через нього може як краще проходити сигнал так і повністю глушитися навіть з однією перешкодою.

Другим етапом експерименту є перевірка швидкості передачі даних на зазначених дистанціях. Отримані дані були занесені у таблицю 4.2 та представлені на рисунку 4.8 у вигляді графіку.

Таблиця 4.2 – Значення швидкості передачі даних в залежності від відстані та перешкод

Дальність, м	Кількість перешкод		
	Без перешкод	Одна стіна	Дві стіни
0	57 мс	65 мс	71 мс
5	117 мс	160 мс	234 мс
10	130 мс	294 мс	312 мс
15	157 мс	348 мс	457 мс

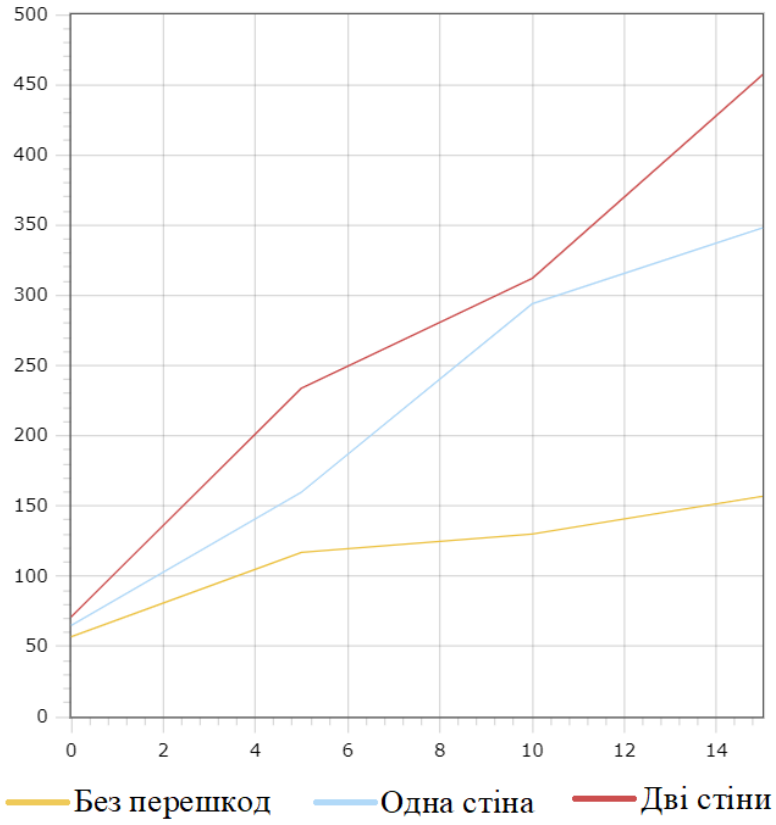


Рисунок 4.8 – Графік залежності швидкості передачі даних до відстані та перешкод

З отриманого даних можна зробити висновок, що відстань та наявність перешкод впливають на швидкість передачі даних, але на подібній дистанції, яка досліджується у цій роботі, втрати швидкості передачі даних які складають менше декількох секунд не є суттєвими.

Останнім етапом експериментального дослідження є передача даних з модуля моніторингу до модуля візуалізації з відстеженням часу необхідного для відображення даних з датчика на розробленому HMI. Для цього до датчика температури Temperature sensor 1 було наближено гарячий об’єкт та замірено час після якого було помічено зміну у графіку. На рисунку 4.9 зображено значення графіків до початку експерименту.

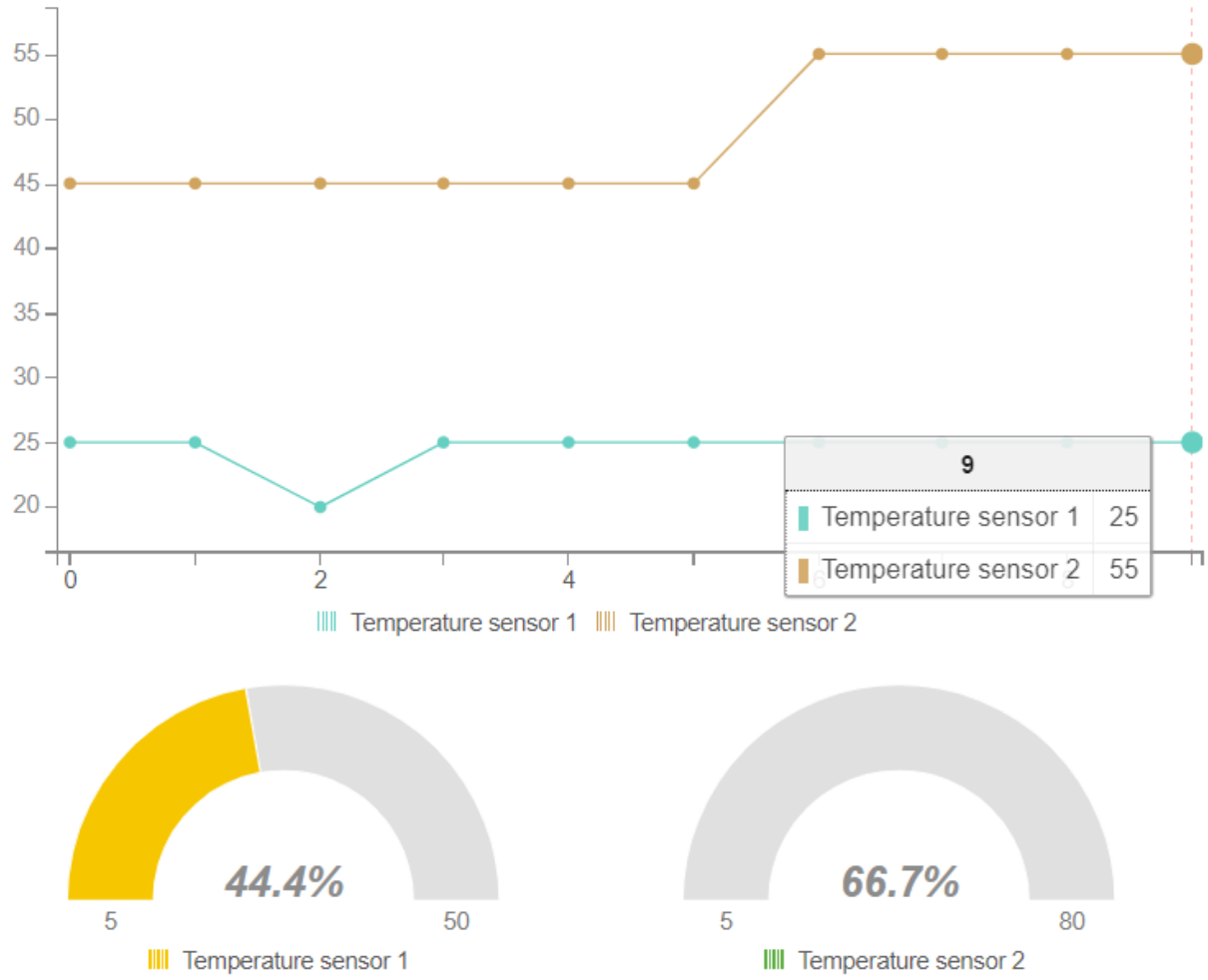


Рисунок 4.9 – Значення графіків HMI до початку експерименту

Після отримання даних з датчика графік Temperature sensor 1 змінив своє значення. На рисунку 4.10 показано значення графіків після отримання даних у ході експерименту

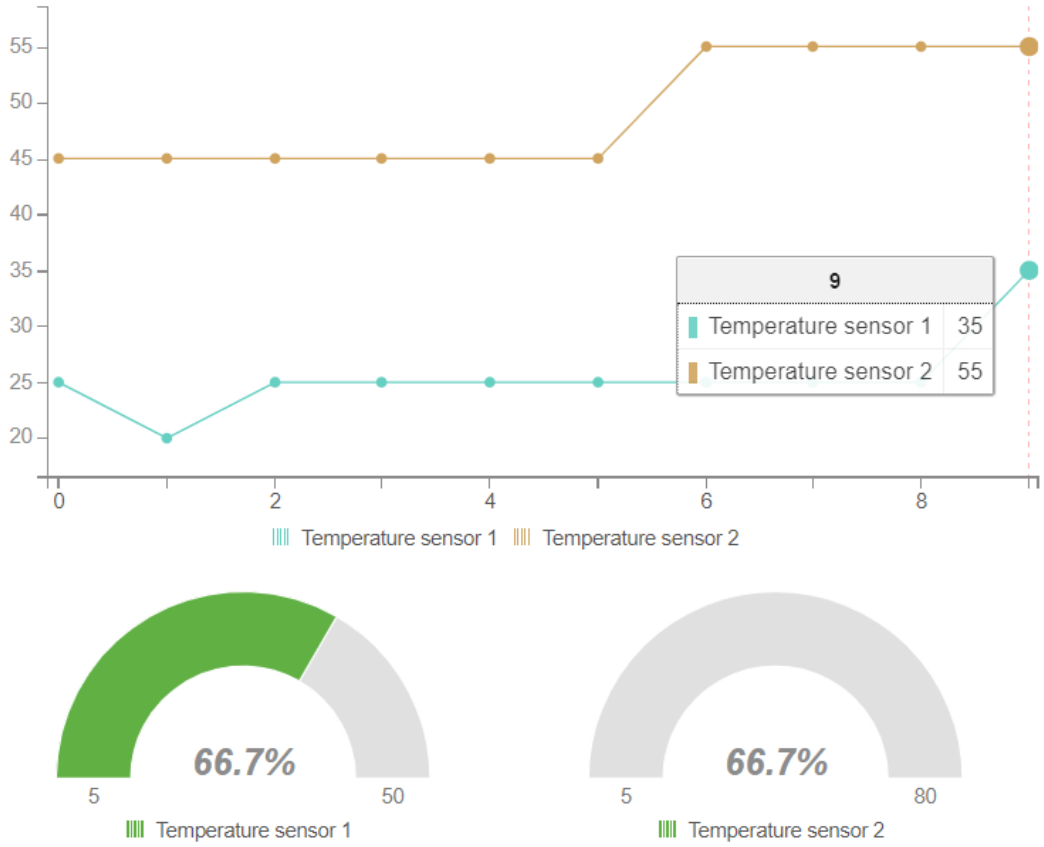


Рисунок 4.10 – Значення графіків HMI після початку експерименту

В ході вимірювання часу було виявлено, що на оновлення графіків потрібно від 5 с до 11 с. Подібна затримка обумовлена циклом отримання даних з бази даних. Також було виявлено, що відстань між модулями не впливає на діапазон часу необхідного для оновлення даних у графіку. Тобто на зазначених вище відстанях оновлення графіку також відбувалося у період від 5 с до 11 с.

4.3 Розрахунок освітленості робочого приміщення і робочого місця для забезпечення безпечних умов роботи

У якості основи для розрахунку необхідного освітленості приміщення для неперервної роботи системи, було обрано приміщення з геометричними розмірами 250 м × 150 м × 20 м.

Нормована освітленість для виробництв подібного розміру складає 500 лк. Напруга в мережі підприємства – 220 В, передбачається використовувати світильники люмінесцентні ЛПО, для даних світильників коефіцієнт використання світлового потоку – 49 %. Відбивна здатність для різних поверхонь:

– стелі 0,7;

– стін 0,5;

– робочої поверхні 0,3.

Робоча поверхня КРЛ розміщена на висоті 0,8 м, висота схилу – 0,1 м. Площа ділянки становить 37500 м^2. Індекс приміщення розраховується формулою (4.3) [25]:

$$\varphi = (a \cdot b\ ) / H_P\ (a\ +\ b), \tag{4.3}$$

де $a,\ b$ – довжина і ширина приміщення;

H_P – висота установки світильників над розрахунковою площиною.

За розрахункову площину було використану висоту письмового столу, тобто $H_2 = 0{,}8$ м. Висота установки світильників розраховується за формулою (4.4):

$$H_P = H_1 -\ H_2, \tag{4.4}$$

У нашому випадку висота установки світильника над рівнем підлоги дорівнює 20 метрам, тобто $H_1 = 20$ м. З цього отримано, що висота установки світильників дорівнює:

$$H_P = 20 -\ 0{,}8 = 19{,}2 \text{ м.}$$

Після знаходження висота установки світильників можливо розрахувати

індекс приміщення, який дорівнює:

$$\varphi = (250 \cdot 150)/\ 19{,}2 \cdot\ (250\ +\ 150) =\ 4{,}9.$$

Для знаходженні кількості світильників N, що забезпечують необхідну освітленість, необхідно використовувати цю формулу (4.5):

$$N = E_{ср} Sk/Un\Phi_{Л}, \qquad (4.5)$$

де $\Phi_{Л}$ – світловий потік лампи, є паспортної характеристикою ламп, лм;

n – кількість ламп в світильнику, шт.;

S – площа освітлюваного приміщення, м^2;

U – коефіцієнт використання;

$E_{ср}$ – середню освітленість приміщення;

k – коефіцієнт запасу, $k =\ 1{,}4$ для сухих чистих приміщень.

Обчислення коефіцієнта використання світильника полягає у вирішенні системи лінійних алгебраїчних рівнянь, складених для всіх поверхонь, що відбивають. Згідно системи лінійних алгебраїчних рівнянь для світильника з відбивачем коефіцієнти використання дорівнює $U = 65$.

Знаючі всі параметри можемо обчислити необхідну кількості світильників для забезпечують освітленості приміщення:

$$N = (500\ \cdot 37500 \cdot 1{,}4)/(65 \cdot 6 \cdot 1150)\ = 58{,}5 \text{ шт.}$$

Округливши результат, отримаємо необхідну кількість світильників, що дорівнює 59 шт.

ВИСНОВКИ

У кваліфікаційній роботі розроблено систему моніторингу та візуалізації даних для кіберфізичних виробничих систем, яка має можливість отримувати дані з інтелектуальних датчиків, опрацьовувати їх, передавати між частинами системи та візуалізувати виробничі дані у реальному часі.

Під час виконання роботи була проведено аналіз сучасних систем моніторингу та візуалізацій даних на виробництві та проаналізовані існуючих SCADA та SCADA/HMI, що використовуються на виробництвах. Виходячи з отриманих даних було вирішено, що кваліфікаційна робота буде спрямована на розробку система моніторингу та візуалізації даних для кіберфізичних виробничих систем особливістю якої є легка інтеграція нових частин SCADA, легкий обмін даними між частинами системи та простота впровадження системи у виробництві.

Після аналізу сучасних систем моніторингу та візуалізацій даних та існуючих SCADA та SCADA/HMI, була розроблена структурна схема, інформаційна модель та архітектура розроблюваної системи. Були проаналізовані та обрані апаратні модулі для розроблюваної системи.

Після розробки структури та інформаційної моделі системи, була обрана середа програмування, розроблені загальний алгоритм роботи системи, логічну та фізичну моделі баз даних. Наступним етапом було виконання компонування пристрою та реалізація алгоритму системи у програмі. Для подальшої роботи обрано інструмент для візуалізації даних у системі.

Останнім етапом було експериментальне дослідження та перевірка працездатності розробленої системи, для чого виконано розробку макету та експеримент в ході якого було досліджено вплив відстані між модулями та наявності перешкод на роботу системи. Крім того було проведено розрахунок освітленості робочого приміщення і робочого місця для забезпечення

безпечних умов роботи.

Завдяки високій актуальність теми залишається великий простір для модернізації макету системи стеження шляхом додання покращень:

– використання інтелектуальних датчиків;

– модифікація системи для додавання автономної роботи;

– додавання та покращення індикаторів у модулі візуалізації;

– покращення швидкості роботи системи за рахунок оптимізації роботи баз даних.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ДСТУ 3008: 2015. Інформація та документація. Звіти у сфері науки і техніки. Структура і правила оформлення. К.: ДП “УкрНДНЦ”. 2016. 30 с.

2. Методичні вказівки з підготовки та захисту кваліфікаційної роботи здобувачами другого (магістерського) рівня вищої освіти спеціальності 151 Автоматизація та комп’ютерно-інтегровані технології, освітньо-професійних програм: «Автоматизоване управління технологічними процесами», «Комп’ютерно-інтегровані технологічні процеси і виробництва», «Комп’ютеризовані та робототехнічні системи»: Навч. посібник / Упоряд. І. Ш. Невлюдов та ін. Харків: ХНУРЕ. 2021. 55 с.

3. Yevsieiev V. Visual components formal description development for the automated design of software products and modules for computer-integrated production technological preparation systems. Вчені записки таврійського національного університету імені В.І. Вернадського Серія: Технічні науки. 2018. Том 29 (68) № 1 Частина 1. С. 143–147. DOI: 10.31474/2075-4272-2018-1-31-24-31.

4. Шалько Є.В. Система стеження і підрахунку об'єктів складної геометричної форми на виробництві з використанням інфрачервоних датчиків. Збірник студентських наукових статей «Автоматизація та приладобудування» / Упоряд.: І.Ш. Невлюдов. Харків: ХНУРЕ. 2021. С. 279 - 283.

5. Understanding Industry 4.0 [Електронний ресурс]. URL: https://ottomotors.com/blog/understanding-industry-4-0 (дата звернення: 18.09.2022).

6. Шалько Є.В. Дослідження застосування протоколу m2m в кібер-фізичних системах COLLECTION OF STUDENTS' SCIENTIFIC PAPER «AUTOMATION AND DEVELOPMENT OF ELECTRONIC DEVICES» ADED-2021 Part 2 (Key infrastructure 2021) - Kharkiv/ The Editorial.:

Nevlyudov I.Sh. (head), that all. Kharkiv: Kind of Kharkiv National Universitety of Radio Elektronics [electronic edition]. 2021. P. 195-199.

7. Yevsieiev V., Bronnikov A. (2020). Analysis of the cyber-physical production systems implementation impact to achieve the goals of lean production. The IIth International scientific and practical conference «Development of scientific and practical approaches in the era of globalization», P. 221–226, DOI:10.46299/ISG.2020.II.II.

8. Syed Imran Shafiq, Edward Szczerbicki, Cesar Saninc. (2019). Proposition of the methodology for Data Acquisition, Analysis and Visualization in support of Industry 4.0. Procedia Computer Science, Volume 159, P. 1976– 1985, DOI:10.1016/j.procs.2019.09.370

9. Sebastian Thiede, Artem Turetskyy, Arno Kwade, Sami Kara, Christoph Herrmann. (2019). Data mining in battery production chains towards multi-criterial quality prediction. CIRP Annals, Volume 68, Issue 1, P. 463–466, DOI:10.1016/j.cirp.2019.04.066.

10. Guejong Jo, Su-Hwan Jang, Jongpil Jeong. (2019). Design and Implementation of CPPS and Edge Computing Architecture based on OPC UA Server. Procedia Computer Science, Volume 155, P. 97–104, DOI:10.1016/j.procs.2019.08.017.

11. Malhotra J., Iqbal F., Sahu A.K., Jha S. (2019) A Cyber-Physical System Architecture for Smart Manufacturing. Advances in Forming, Machining and Automation. Lecture Notes on Multidisciplinary Industrial Engineering, P. 637–647, DOI:10.1007/978-981-32-9417-2_53.

12. Christine Schulze, Sebastian Thiede, Bastian Thiede, Denis Kurle, Stefan Blume, Christoph Herrmann. (2019). Cooling tower management in manufacturing companies: A cyber-physical system approach. Journal of Cleaner Production, Volume 211, P. 428–441, DOI:10.1016/j.jclepro.2018.11.184.

13. R. van de Sand, S. Schulz, J. Reiff-Stephan. (2019). Smart Process

Communication for Small and Medium-Sized Enterprises. Proceedings of the I-ESA Conferences, Vol. 9, P. 411−420, DOI:10.1007/978-3-030-13693-2_34.

14. Hammer M. (2019) Digitization Perspective: Impact of Digital Technologies in Manufacturing. In: Management Approach for Resource-Productive Operations. Industrial Management, P.27−68, DOI:10.1007/978-3-658-22939-9_3.

15. Chiara Cimini, Fabiana Pirola, Roberto Pinto, Sergio Cavalieri. (2020). A human-in-the-loop manufacturing control architecture for the next generation of production systems. Journal of Manufacturing Systems, Volume 54, P. 258−271, DOI:10.1016/j.jmsy.2020.01.002.

16. Majid Al-Kuwari, Abdulrhman Ramadan, Yousef Ismael, Laith Al-Sughair, Adel Gastli, and Mohieddine Benammar. ”Smart-Home Automation using IoT-based Sensing and Monitoring Platform”. 2018 IEEE 12th International Conference on Compatibility, Power Electronics and Power Engineering (CPE-POWERENG 2018). 2018. P. 1−6.

17. Rohini Shete and Sushma Agrawal, ”IoT Based Urban Climate Monitoring using Raspberry Pi” 2016 International Conference on Communication and Signal Processing (ICCSP). 2016. P. 2008−2012.

18. Milica Lekic, and Gordana Garadasevic, “IoT Sensor Integration to Node- RED Platform”, 17th International Symposium INFOTEH-JAHORINA. 21-23 March 2018. P. 1–5.

19. Anoja Rajalakshmi, and Hamid Shahnasser. ”Internet of Things using Node- Red and Alexa”. 2017 17th International Symposium on Communications and Information Technologies (ISCIT). 2017. P. 1–4

20. Security issues in SCADA networks Vinay M. Igure*, Sean A. Laughter, Ronald D. Williams Charles L. Brown Department of Electrical and Computer Engineering, University of Virginia, Charlottesville, VA 22904, USA Computers & Security 25 (2006). P.498–506.

21. Shalko Y. Analysis of Production Data Monitoring and Visualization Systems for Cyber-Physical Production Systems. International Conference «Manufacturing & Mechatronic Systems 2022» / Упоряд.: І.Ш. Невлюдов. Харків: ХНУРЕ. 2022. С. 39 – 42.

22. Understand Architecture and Block Diagram of SCADA System [Електронний ресурс]. URL: https://www.etechnog.com/2021/11/architecture-block-diagram-scada-system.html (дата звернення : 02.10.2022).

23. Compare - SCADA Software [Електронний ресурс]. URL: https://www.goodfirms.co/scada-software (дата звернення : 14.10.2022).

24. The New SCADA [Електронний ресурс]. URL: https://www.inductiveautomation.com/resources/article/old-scada-vs-the-new-scada (дата звернення : 19.10.2022).

25. COM-Based SBCs: The Superior Architecture for Small Form Factor Embedded Systems [Електронний ресурс]. URL: https://whitepaper.opsy.st/WhitePaper.diamondsys-combased-sbcs-wpfinal-.pdf (дата звернення : 22.11.2022).

26. Raspberry Pi Zero 2 W [Електронний ресурс]. URL: https://datasheets.raspberrypi.com/rpizero2/raspberry-pi-zero-2-w-product-brief.pdf (дата звернення : 25.11.2022).

27. Raspberry Pi 4 Tech Specs [Електронний ресурс]. URL: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/ (дата звернення : 25.11.2022).

28. LattePanda 3 Delta [Електронний ресурс]. URL: https://www.lattepanda.com/lattepanda-3-delta (дата звернення : 25.11.2022).

29. Що таке інтелектуальний датчик? - визначення з техопедії - Обладнання 2022 [Електронний ресурс]. URL: https://uk.theastrologypage.com/intelligent-sensor (дата звернення : 25.11.2022).

30. Arduino Mega 2560 Rev3 [Електронний ресурс]. URL:

https://store.arduino.cc/products/arduino-mega-2560-rev3 (дата звернення: 25.11.2022).

31. PhpStorm: PHP IDE and Code Editor from JetBrains [Електронний ресурс]. URL: https://www.jetbrains.com/phpstorm/ (дата звернення : 27.11.2022).

32. Вибір технологій та баз даних для розробки корпоративного ПЗ [Електронний ресурс] . URL: https://almexecm.com.ua/novosti-i-sobyitiya/vybir-tehnologij-ta-baz-danyh-dlya-rozrobky-korporatyvnogo-pz.html (дата звернення : 27.11.2022).

33. The application/json Media Type for JavaScript Object Notation (JSON) [Електронний ресурс]. URL: https://www.rfc-editor.org/rfc/rfc4627 (дата звернення : 28.11.2022).

34. Chapter 7: Distributed Application Architecture [Електронний ресурс]. URL: https://web.archive.org/web/20110406121920/http://java.sun.com/developer/Books/jdbc/ch07.pdf (дата звернення : 29.11.2022).

35. Zero vs Thin vs Thick Clients: What's Right for Your Business? [Електронний ресурс] – URL: https://biztechmagazine.com/article/2018/10/thin-vs-thick-vs-zero-client-whats-right-fit-your-business-perfcon (дата звернення : 29.11.2022).

36. billboard.js - examples [Електронний ресурс]. URL: https://naver.github.io/billboard.js/demo/ (дата звернення : 30.11.2022).