МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М. Є. Жуковського
«Харківський авіаційний інститут»

Факультет програмної інженерії та бізнесу

Кафедра інженерії програмного забезпечення

Пояснювальна записка до дипломної роботи

магістра
(освітній ступінь)

на тему: «Розробка програмно-апаратної системи вібраційної діагностики промислового обладнання. Серверна частина»

ХАІ.603.667п2.121.166380.20О

Виконав: студент 6 курсу групи 667п2
(прізвище й ініціали студента)
Спеціальність 121 – Інженерія програмного забезпечення
(код та найменування)
Освітня програма Хмарні обчислення та Інтернет речей
(найменування)
Лезновський В.А.
(прізвище й ініціали студента)
Керівник: Сергієнко В.В
(прізвище й ініціали)
Рецензент: _____
(прізвище й ініціали)

Харків – 2020

Міністерство світи і науки України
Національний аерокосмічний університет ім. М. Є. Жуковського
«Харківський авіаційний інститут»

Факультет програмної інженерії та бізнесу
(повне найменування)
Кафедра інженерії програмного забезпечення
(повне найменування)
Рівень вищої освіти другий (магістерський)
Спеціальність 121 – інженерія програмного забезпечення
(код та найменування)
Освітня програма хмарні обчислення та Інтернет речей
(найменування)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Туркін І.Б.
(підпис) (ініціали та прізвище)

“_____” _____ 2020 року

З А В Д А Н Н Я
НА ДИПЛОМНУ РОБОТУСТУДЕНТУ

Лезновському Вячеславу Андрійовичу
(прізвище, ім’я, по батькові)

1. Тема дипломної роботи Розробка програмно-апаратної системи вібраційної діагностики промислового обладнання.Серверна частина
керівник дипломної роботи Сергієнко В.В, к.т.н, доцент
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом Університету № _____ від “____” _____ 2020 року
2. Термін подання студентом роботи _____
3. Вихідні дані до роботи - Наукові статті з оцінки алгоритмів машинного навчання за умов багатокритеріальності
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
1) Існуючий стан вібраційної діагностики
2) Основні параметри системи вібраційної діагностики
3) Розробка прототипу програмного забезпечення системи вібраційної системи
5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
Пояснювальна записка містить 74 сторінки, 54 рисунків, 2 таблиці, 11 джерел.

1) Для першого розділу розроблено 4 слайдів презентації, 6 рисунків та 0 таблиць.
2) Для другого – 7 слайда презентації, 31 рисунків та 0 таблиць.
3) Для третього – 4 слайдів презентації, 18 рисунків та 2 таблиці.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада Консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Сергієнко В.В., доцент.		
2	Сергієнко В.В., доцент.		
3	Сергієнко В.В., доцент.		

Нормоконтроль _____ _____ «___»_____20__р.
(підпис) (ініціали та прізвище)

7. Дата видачі завдання «01»___09____2020__ р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного роботи	Строк виконання етапів роботи	Примітка
1	Отримання і затвердження теми диплому	01.09.2020	
2	Обґрунтування актуальності дослідження ефективності використання систем	12.09.2020	
3	Формулювання мети, об’єкту, предмету та методів дослідження	15.09.2020	
4	Огляд існуючих аналогів, пошук недоліків у аналогів	20.09.2020	
5	Розробка архітектури та інтерфейсу	01.10.2020	
6	Розробка прототипу ПЗ	10.10.2020	
7	Планування експерименту	21.10.2020	
8	Виконання експерименту систем	05.11.2020	
9	Підготовка пояснювальної записки	20.11.2020	
10	Оформлювання пояснювальної записки до дипломної роботи	22.11.2020	
11	Передзахист дипломної роботи	30.11.2020	
12	Захист дипломної роботи	14.12.2020	

Студент _____ Лезновський В.В
(підпис) (прізвище та ініціали)

Керівник роботи _____ Сергієнко В.В.

РЕФЕРАТ

Пояснительная записка на дипломную работу: 83 с., 54 ил., 11 источников.

Тема работы - разработка программно-аппаратной системы вибродиагностики промышленного оборудования. Серверная часть.

Цель дипломной работы магистра - повышение эффективности эксплуатации оборудования на промышленных предприятиях путем внедрения беспроводной системы вибродиагностики.

Для достижения цели необходимо выполнить следующие задачи:

1) провести анализ методов оценки состояния оборудования;
2) провести анализ имеющихся технических решений диагностики оборудования;
3) разработать систему вибрационной диагностики и оценки состояния оборудования;
4) выполнить планирование эксперимента для исследования системы вибрационной диагностики оборудования;
5) разработать алгоритмы обработки результатов эксперимента для исследования системы вибрационной диагностики оборудования;
6) на основе анализа результатов эксперимента сформулировать практические рекомендации применения разработанных методов и программно технических решений для оценки оборудования с помощью вибрационной диагностики.

Актуальность задачи - создание информационной системы, которая будет полезна для анализа состояния оборудования на предприятии, а также повысит безопасность эксплуатации оборудования и снизит количество простоев оборудования.

Основным практическим результатом работы является разработка инструментальных средств вибрационного диагностики, а также стенда для верификации разработанной системы.

Необходимо разработать методы и модели которые станут основой для развития программно-аппаратных систем вибрационного диагностики и разработки встроенного программного обеспечения.

ОСНОВНОЙ ЦЕЛЬЮ ВНЕДРЕНИЯ ПРОГРАММНОГО-АППАРАТНОГО КОМПЛЕКСА ЯВЛЯЕТСЯ ПОВЫШЕНИЕ ЭФФЕКТИВНОСТИ ЭКСПЛУАТАЦИИ ОБОРУДОВАНИЯ НА ПРОМЫШЛЕННЫХ ПРЕДПРИЯТИЯХ.

РЕФЕРАТ

Пояснювальна записка на дипломну роботу: 83 с., 54 іл., 11 джерел.

Тема роботи - розробка програмно-апаратної системи вібродіагностики промислового обладнання. Серверна частина.

Мета дипломної роботи магістра – підвищення ефективності експлуатації обладнання на промислових підприємствах шляхом використання бездротової системи вібродіагностики.

Для досягнення мети необхідно виконати наступні завдання:

1) провести аналіз методів оцінки стану обладнання;
2) провести аналіз наявних технічних рішень діагностики обладнання;
3) розробити систему вібраційної діагностики та оцінки стану обладнання;
4) виконати планування експерименту для дослідження системи вібраційної діагностики обладнання;
5) розробити алгоритми оброблення результатів експерименту для дослідження системи вібраційної діагностики обладнання;
6) на основі аналізу результатів експерименту сформулювати практичні рекомендації застосування розроблених методів та програмно технічних рішень для оцінки обладнання за допомогою вібраційної діагностики.

Актуальність завдання - створення інформаційної системи яка буде корисна для аналізу стану обладнання на підприємстві, а так само підвищить безпеку експлуатації устаткування і знизить кількість простоїв обладнання.

Основним практичним результатом роботи програмного комплексу є розробка інструментальних засобів вібраційного діагностики а так же стенду для верифікації розробленої системи.

Необхідно розробити методи і моделі, що є основою для розвитку програмно-апаратних систем вібраційного діагностики та розробки вбудованого програмного забезпечення.

ОСНОВНОЮ МЕТОЮ ВПРОВАДЖЕННЯ ПРОГРАМНОГО-АПАРАТНОГО КОМПЛЕКСУ Є ПІДВИЩЕННЯ ЕФЕКТИВНОСТІ ЕКСПЛУАТАЦІЇ ОБЛАДНАННЯ НА ПРОМИСЛОВИХ ПІДПРИЄМСТВАХ.

ABSTRACT

Explanatory note for thesis: 83 p., 54 ill., 11 sources.

Work theme- development of a software and hardware system for vibrodiagnostics of industrial equipment. Server side.

The goal of the master's degree project is to increase the efficiency of equipment operation at industrial enterprises by introducing a wireless vibration diagnostics system.

To achieve the goal, you must complete the following tasks:

1) to analyze the methods of assessing the condition of the equipment;
2) to analyze the available technical solutions for equipment diagnostics;
3) to develop a system of vibration diagnostics and equipment condition assessment;
4) perform experimental planning to study the system of vibration diagnostics of equipment;
5) to develop algorithms for processing the results of the experiment to study the system of vibration diagnostics of equipment;
6) on the basis of the analysis of results of experiment to formulate practical recommendations of application of the developed methods and software and technical decisions for an estimation of the equipment by means of vibration diagnostics.

The urgency of the task is to create an information system that will be useful for analyzing the state of equipment at the enterprise, as well as increase the safety of equipment operation and reduce the number of equipment downtime.

The main practical result of the software complex is the development of vibration diagnostics tools as well as a stand for verification of the developed system.

It is necessary to develop methods and models, which is the basis for the development of software and hardware systems for vibration diagnostics and the development of embedded software.

THE MAIN PURPOSE OF THE IMPLEMENTATION OF THE SOFTWARE AND HARDWARE COMPLEX IS TO INCREASE THE EFFICIENCY OF EQUIPMENT OPERATION AT INDUSTRIAL ENTERPRISES.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ ТА ТЕРМІНІВ

БД – база даних;

ОС – операційна система;

ПК – персональний комп’ютер;

ПЗ – програмне забезпечення;

Пакет – в телекомунікаціях це відформатований блок даних який передається за допомогою комутації пакетів в комп'ютерній мережі;

СУБД – система управління базою даних;

Фреймворк – вузько направлена бібліотека для вирішення завдань;

IoT – це мережа пов'язаних через інтернет об'єктів, здатних збирати дані і обмінюватися даними, які надходять зі вбудованих сервісів;

MVC – архітектурний шаблон «Модель–вигляд–контролер».

ВСТУП

Вібраційна діагностика важливий спосіб оцінки стану обладнання зокрема обертового. Підвищення надійності і ефективності обертового обладнання - центральна проблема в багатьох областях застосування, такі як виробництво і транспортування енергії. Це вимагає ефективних методів моніторингу стану, включаючи аналітику для прогнозування і виявлення несправності і можливість управляти великими обсягами даних. Обертові елементи підшипників є важливими компонентами обертових машин і особливо важливі для моніторингу в зв'язку з жорсткими вимогами до умов їх експлуатації. Крім того, підшипники розташовані поруч з обертовими частинами машин і тим самим сигналізують джерела, що характеризують несправності і ненормальні умови експлуатації.

Науково-прикладна задача – розробити систему вібраційної діагностики з використанням методів, алгоритмів та технічних рішень для оцінки стану обладнання.

Метою роботи є підвищення ефективності роботи обертового обладнання за допомогою оперативної вібраційної діагностики з використанням MEMS акселерометрів. Для досягнення мети необхідно виконати наступне завдання – розробити програмно-апаратний комплекс вібраційної діагностики для оперативної оцінки стану обладнання.

Завдання для досягнення мети:

1) провести аналіз методів оцінки стану обладнання;
2) провести аналіз наявних технічних рішень діагностики обладнання;
3) розробити систему вібраційної діагностики та оцінки стану обладнання;
4) виконати планування експерименту для дослідження системи вібраційної діагностики обладнання;
5) розробити алгоритми оброблення результатів експерименту для дослідження системи вібраційної діагностики обладнання;
6) на основі аналізу результатів експерименту сформулювати практичні рекомендації застосування розроблених методів та програмно технічних рішень для оцінки обладнання за допомогою вібраційної діагностики.

Об’єктом дослідження є оцінка стану обладнання за допомогою вібраційної діагностики.

Предметом дослідження є метод оцінки стану обладнання за допомогою аналізу вібраційних показників.

Методи дослідження:

1) аналіз – метод буде використовуватися для виділення основних ознак при класифікації дефектів які призводять до некоректної роботи обертового обладнання;
2) порівняння – метод буде використовуватися при визначенні класів пошкоджень та їх розмежуванні;
3) вимірювання – процедура визначення чисельного значення певної величини за допомогою одиниці виміру. Цей метод буде використовуватися для оцінювання точності розроблюваної системи;

4) експеримент – метод буде використовуватися при моделюванні;
5) матеріальне моделювання – метод що використовується при моделюванні роботи обертового обладнання за допомогою стенду.

Наукова новизна - удосконалений метод вібраційної діагностики для автоматизованого оцінювання стану обладнання.

Практична значущість отриманих результатів полягає в тому що впровадження подібної системи дозволить користувачам зменшити витрати на проведення вібраційного діагностики, а також змінити модель обслуговування обладнання на «обслуговування за реальним станом обладнання». При цій стратегії обсяг ремонтів і час між ними заздалегідь не фіксовані, а визначаються за результатами регулярних ревізій (обстежень) обладнання і моніторингу його стану за допомогою автоматизованих засобів контролю і діагностики. Ця стратегія дозволяє істотно економити ресурси, тому вона вважається найбільш прогресивною для складного і дорогого обладнання. Ефективність ТОіР визначається відношенням максимально можливого результату ТОіР (висока якість робіт при дотриманні нормативного терміну ремонту) до мінімально можливим експлуатаційних витрат (мінімально обґрунтований рівень витрат без втрати якості і обсягу виконаних робіт). Вібраційна діагностика є найбільш ефективною при оцінці стану обладнання з рахунком цього система що дозволяє вести постійну діагностику з автоматичною класифікацію стану обладнання є практично значущою.

1 АНАЛІЗ СУЧАСНОГО СТАНУ ПРОБЛЕМИ ВІБРАЦІЙНОЇ ДІАГНОСТИКИ. ПОСТАНОВКА ЗАДАЧ ДИПЛОМНОЇ РОБОТИ

Інженери та дослідники, особливо в аерокосмічної і морський нафтовий промисловості, почали використовувати виявлення пошкоджень на основі вібрації за часів кінця 1970-х - початку 1980-х років . Ці підходи були засновані на кореляції чисельних моделей з вимірюваними модальними властивостями з неушкоджених і пошкоджених компонентів. Також подібні методи використовувалися в морської нафтогазової промисловості, включаючи дослідницькі цілі для виявлення близького до відмови бурового обладнання і запобігання поломки дорогих масляних насосів . Істотні практичні проблеми включали вплив шуму платформи на вимірювання, проблеми з обладнанням у ворожому середовищі, різні ефекти навантаження на бурильної трубі, зміна маси платформи що викликана зростанням рівня моря. Ці труднощі заважали розвитку методів діагностики що основані на частотному аналізі та призвели до того що с 1980х років дослідження цих методів значно зменшились хоча деякі дослідники продовжують просувати ці методи. Згідно Фаррар і Доблінгу , найбільш зріле і успішне застосування технології виявлення пошкоджень на основі вібрації було в моніторингу обертового обладнання. Методологія виявлення заснована на зразку. Розпізнавання може бути застосовано до часової а також частотної області. Бази даних дозволяють визначати певні типи пошкоджень. Вони визначаються по виміряним сигнатурам вібрації. Такий тип моніторингу не використовується для великих структур на які впливають багато факторів довкілля. Дані в часовій області можуть вимірюватися різними датчиками, наприклад акселерометри, тензодатчики і т. д., і ці дані можуть потім можна перетворити з тимчасової області в частотну область з використанням перетворення Фур'є. Подальший аналіз даних частотної області часто використовується для вилучення модальних параметрів для створення так званих модальних даних. Також існують методи прямого перетворення даних.

Вимірювання завжди виробляються в часовій області, аналітик з моніторингу стану може вибрати аналіз даних у часі, частоті або модальні дані. Хоча перетворення між областями вимагає деякого стиснення даних, Фрісвелл і Пенні стверджують, що для лінійних систем втрата інформації невелика. Крім того, існує деяка перевага в тому, що дані можна легко усереднити і таким чином можна зменшити вплив випадкового шуму. Модальна область передбачає подальше скорочення обсягу даних у порівнянні з частотою домен. Більшість літератури на сьогодні описує методи засновані на модальному аналізі. Це пов’язано з тим що рання література була зосереджена на модальному аналізу. Дослідження методів, заснованих на всіх трьох областях продовжаться, в першу чергу тому, що ще не знайдено жодного методу, який би визначає кожен тип пошкодження в кожному типі структури. В результаті класифікації проблем у різних структурах та системах, значні дослідницькі зусилля були застосовані до розробки широкого кола методів та алгоритмів моніторингу стану обладнання. Ріттер класифікував різні методи на основі рівня ідентифікації:

1) визначення наявності пошкодження у структурі;
2) визначення геометричного розташування пошкодження;
3) кількісна оцінка тяжкості пошкодження;
4) прогнозування терміну служби, що залишився.

Багато методів ідентифікації пошкоджень таких як методи обновлення матриці описані у літературі присвяченій FEM методам. Більшість методів які не використовують FEM методи, опираються на кореляційний чисельний аналіз який було описано вище.

Хоча реальні пошкодження конструкції можуть бути локальними або розподіленими, методи FEM як правило більш підходять до розподілених пошкоджень. Використання великої кількості параметрів пошкоджень вкупі з обмеженою кількістю вимірюваних показників може привести до труднощів у використанні FEM алгоритмів.

Методи виявлення пошкоджень також різняться по кількості датчиків необхідних для роботи методів. Резонансні частоти можуть бути виміряні з використанням одного або декількох датчиків, тоді як визначення форми або динамічна гнучкість вимагають декількох датчиків. Таким чином, виникають питання економічної доцільності, особливо якщо намір полягає в тому, щоб контролювати мережу структур. Деякі з найбільш успішних результатів виявлення пошкоджень були досягнені в лабораторних умовах, де відсутність обертальних ступенів свобод) і легкість доступу дозволяють виміряти форми коливань[1]. Однак навіть з великою кількістю датчиків вимірювання модальних характеристик зазвичай є неповним.

У літературі основна увага приділяється рівню з 1 по 3. Велика частина літератури присвячена лабораторним конструкціям або контрольованим пошкодженням структурам і не намагаються передбачити термін служби структури. 4 рівень безсумнівно є кінцевою метою будь-якої системи моніторингу стану. Класифікація різноманітних методів виявлення пошкоджень наступна:

1) методи засновані на аналізу резонансної частоти;
2) методи засновані на формі сигналу;
3) методи засновані на аналізу форми кривизни \ аналіз форми деформації;
4) методи засновані на динамічному вимірювані гнучкості;
5) FEM методи;
6) нелінійні методи;
7) методи засновані на нейронних мережах.

1.1 Базові методи аналізу стану обладнання за допомогою вібраційної діагностики

1.1.1 Методи засновані на аналізі резонансних частот

Фізичний зв'язок між жорсткістю, зміною маси та резонансною частотою в поєднанні з простотою вимірювання резонансної частоти в багатьох випадках потрібен тільки один датчик) став поштовхом для використання модальних методів виявлення пошкоджень.

В роботі Салава розглянуті публікації зв'язані з виявленням структурних пошкоджень за допомогою вимірювання зміни частот [10]. Більшість ранніх робот були сконцентровані на простих структурах або простих структурних елементах. Вимірювання однієї пари частот може показати місце можливого пошкодження. Місця деяких пар частот можуть накладатися, фактичне місце пошкодження задається перетином кривих. Додатково Бенкс і ін. [2] показали, що геометрія ушкодження, а не тільки місцезнаходження і серйозність, впливає на резонансні частоти. Наприклад оброблені пази в випробувальних зразках призводять до зміни резонансних частот реальних тріщин.

Кесслер та ін. [3] вивчають вплив на АЧХ різних форм ушкоджень (Просвердлені наскрізні отвори, удари, тріщини, вигини і втома матеріалів). На затиснутих композитних пластинах і прийшов до висновку, що єдиний тип ушкоджень що відрізняється від інших в низькочастотних діапазонах були пошкодження від втоми. Це було через те що пошкодження викликані втомою матеріалу виробляють багато високоенергетичних локальних максимумів яких не було в інших видах пошкоджень.

З огляду на те що зміна частоти через випадкову \ навколишню вібрацію і впливу навколишнього середовища може досягати 5-10%, тому у роботі було визнано що низькочастотні зміни будуть не дуже корисні для ідентифікації пошкоджень. Незважаючи на вищесказане, значна група інших дослідників підтримує використання модальних частот для ідентифікації пошкоджень.

Можливо також використовувати 3D-графіки зміни частоти в залежності від глибини і місця розташування тріщини, щоб визначити пропил в алюмінієвій балці. Перетин контурних ліній що були отримані з кожного графіка зміни частоти показував розташування і глибину тріщини. Виявлення множинних ушкоджень з використання змін частот навіть для простих лабораторних структур не так ефективно.

Спроби використання здвигів частот для виявлення пошкоджень були численними хоча і непереконливими. Успішні алгоритми ідентифікації зазвичай були обмежені ідентифікацією одного або декількох пошкоджень, а також примінились до невеликих лабораторних структур. Використання тільки частотних здвигів в реальних системах моніторингу на сьогодні не здається багатообіцяючою.

1.1.2 Методи засновані на аналізу форми тональності

Для виявлення тональності доступно багато методів модального аналізу. які працюють з даними у часовій області.

1.1.3 Пряме порівняння форм тональності

Два найбільш часто використовуваних методу порівняння для порівняння форм тональності це Modal Assurance Criterion, MAC та Coordinate Modal Assurance Criterion, COMAC. Значення MAC можна розглядати як міру подібності двох форм коливань. Значення MAC, рівне 1, є ідеальним збігом, а значення 0 означає, що вони зовсім не схожі. Таким чином, зниження значення MAC може вказувати на пошкодження. Прикладом можна привести тестування моста Хоча перші сім власних частот зрушили менш ніж на 3%, значення MAC показали суттєва зміна, що змусило авторів стверджувати, що порівняння форм коливань це більш надійний метод виявлення пошкоджень, ніж зрушення резонансних частот. COMAC - це точкова міра різниці між двома наборами форм коливань, що приймає значення від 1 до 0. Низьке значення COMAC вказувало б на невідповідність в певній точці і таким чином, це також індикатор можливого місця пошкодження.

Недоліком багатьох методів, заснованих на формі коливань, є необхідність проведення вимірювань у великій кількості місць. Цікавою технікою є використання скануючого лазерного допплерівського віброметра (LDV), який дозволяє отримувати щільну сітку вимірювань. Хан і ін. [4] використовували скануючого LDV для вимірювання форми коливань в стальній консольної балці, сталевий консольної пластині і бетонних балках. Виявилося, що в товстих металевих конструкціях дефекти виявляються тільки тоді, коли вони проходять більш ніж на половину товщини.. Вони прийшли до висновку, що, хоча методи, засновані на формі коливань, добре перевірені за допомогою змодельованих даних, існують значні труднощі з повномасштабними структурами - переважаючими з них є шум і помилки вимірювань.

1.1.4 Методи засновані на аналізу кривизни

Використання кривизни форм коливань при ідентифікації пошкоджень засноване на припущенні, що зміни кривизни форм коливань сильно локалізовані в області пошкодження і що вони більш виражені, ніж зміни зміщення форм коливань, хоча , це не обов'язково так. Модальні кривизни також використовувалися в поєднанні з іншими вимірами даними для виявлення пошкоджень. Найкращі результати були досягнуті при поєднанні кривизни форми коливань і статичних зміщень. При використанні змодельованої форми кривизни балки в алгоритмі поновлення моделі на основі чутливості для виявлення пошкоджень. Було виявлено, що хоча кривизна більш чутлива до пошкоджень ніж форми зміщення, збіжність була покращена додаванням модальної кривизни. Кількість модальних вигинів для проведення процедур ідентифікації пошкоджень, природно, обмежена кількістю доступних форм тональності зміщення. Прагнучи збільшити обсяг даних, доступних для введення в процедури виявлення пошкоджень, поширив метод кривизни на всі частоти в діапазоні вимірювань, використовуючи дані FRF. На закінчення, форми коливань і їх похідні широко використовувалися для визначення пошкоджень. Деякі дані припускають, що методи, засновані на формах коливань, більш надійні, ніж методи, засновані на зрушеннях резонансних частот. Проте, є деякі суперечності з приводу корисності однієї форми коливань для виявлення пошкоджень навіть від тих же авторів. Така невизначеність призвела до дослідження інших методів, таких як використання робочих форм відхилення, які мають багато спільного з формами коливань. Також були досліджені більш складні формулювання, що включають використання форм коливань, такі як методи модальної енергії деформації, в яких використовуються модальні кривизни і динамічної яка вимірюється гнучкість і вектор залишкової сили, які поєднують використання резонансних частот і форм коливань.

1.1.5 Методи засновані на аналізу робочої форми відхилення

Робочі форми прогину, ODS, залежать від місця розташування і відносних величин сил, прикладених до конструкції. Якщо структура порушується в одному місці поблизу резонансу, то форма коливань і ODS будуть аналогічними. ODS забезпечують візуальну інтерпретацію моделей вібрації конструкції. Можливо генерувати експериментальні дані ODS пучка за допомогою п'єзокерамічних приводів і вимірювали їх за допомогою сковуючого лазерного допплерівського віброметра. Місце пошкодження визначалося за наявністю перегину / розриву на графіку ODS. Розташування пошкоджень було простіше, коли місце порушення було ближче до місця пошкодження. ODS був більш чутливий до пошкодження на більш високих частотах, ніж на більш низьких частотах.

1.1.6 Методи засновані на аналізу потенційна енергія деформації

Коли конкретний режим вібрації зберігає велику кількість енергії деформації на конкретному шляху навантаження конструкції, частота і форма цього режиму дуже чутливі до змін в цьому шляху навантаження. Таким чином, зміни в модальної енергії деформації також можуть розглядатися в якості логічного вибору індикатора місця пошкодження. Література в основному зосереджена на методах одновимірного деформування, хоча вони можуть бути застосовані до двовимірним і тривимірним структурам, які можна розкласти на балкові елементи. Кривизна, необхідна для цього розрахунку, зазвичай вилучають із виміряних форм деформації зсуву з використанням апроксимації центральної різниці. Можливо застосувати алгоритм ідентифікації пошкоджень для визначення місця розташування та визначення розміру окремої тріщини в експериментальній пластинчастої балці. Метод також був продемонстрований для виявлення до двох ділянок ушкодження в моделюється пластинчастої балці. Показник пошкодження був заснований на співвідношенні модальної енергії деформації елементів до і після пошкодження. Метод статистичних гіпотез був використаний для класифікації значущості значення показника збитку..

1.1.7 Методи динамічного вимірювання гнучкості на основі матриці

Матриця гнучкості визначається як зворотна матриці жорсткості і, отже, пов'язує прикладену статичну силу з результуючим зміщенням конструкції. Таким чином, кожен стовпець матриці гнучкості представляє зразок зсуву, пов'язаний з одиничною силою, яка додається до відповідного ступеня свободи. Крім того, через зворотній залежності квадрата модальних частот матриця гнучкості найбільш чутлива до змін в низькочастотних областях.

1.1.8 Метод вектора залишкової сили

Маючи доступ до виміряних формам коливань, резонансним частотам і вихідної базової моделі, можна сформулювати те, що відомо як Вектор залишкової сили, RFV. Кожен рядок RFV представляє одну ступінь свободи числовий моделі конструкції. Коли відбувається пошкодження елемента, пов'язаного з цим ступенем свободи, запис в RFV стає дуже великий у порівнянні з іншими записами, де ніяких пошкоджень не відбулося. Таким чином, виділення цих великих термінів дає метод визначення місця пошкодження. Для кількісної оцінки збитку необхідні наступні алгоритми. представив кілька чисельно змодельованих прикладів алгоритму замкнутої форми для ідентифікації пошкоджень з використанням цього підходу. Косматка і Ріклес [5] ідентифікували поодинокі випадки пошкодження (втрата жорсткості, ослаблення з'єднання, додавання маси шматка) в лабораторній фермі з 135 елементів. Виміри проводилися при кожній ступеня свободи для отримання повних форм коливань. RFV використовувався для визначення місця

пошкодження. Алгоритм зваженої чутливості оцінив величину зміни жорсткості / маси. Як і очікувалося, було виявлено, що підвищена кореляція між аналітичною моделлю і базовими модальними властивостями поліпшила оцінки серйозності пошкоджень. При достатніх вимірах RFV здається надійним метод визначення місця пошкодження і його використання для визначення розміру пошкоджень, безумовно, багатообіцяючий

1.1.9 Методи відновлення моделі

Література, присвячена оновленню моделей, надала багате джерело алгоритмів, які можна адаптувати для ідентифікації пошкоджень. Крім того, багато алгоритмів ідентифікації пошкоджень покладаються на добре кореляційну чисельну модель конструкції в її початковому стані. При створенні корелятивною чисельної моделі виникає ряд проблем:

1) вибрані дані вимірювань для зіставлення з моделлю;
2) точність вихідної моделі;
3) розмір і складність моделі;
4) кількість параметрів оновлення;
5) неоднозначність отриманої моделі відповідно до виміряними даними.

Важлива точність вихідної моделі конструкції, використовуваної для виявлення пошкоджень за допомогою алгоритму поновлення. використовували алгоритми, засновані на чутливості, для виявлення і виявлення пошкоджень. Розмір моделі, яка підлягає оновленню, викликає заклопотаність, хоча з постійно збільшується доступною обчислювальною потужністю тепер можна працювати з більшими і складними моделями, ніж будь-коли раніше. Однак, оскільки проблеми оновлення моделі зазвичай вирішуються ітераційним методами, які потребують вирішення аналітичної задачі принаймні один раз на кожній ітерації, його застосування до великих моделям може зажадати значних ресурсів процесора. Теоретична кількість параметрів, одержуваних при зіставленні власних значень, дорівнює кількості виміряних резонансних частот. При використанні форм коливань кількість параметрів має верхню межу, що дорівнює кількості режимів, помноженому на кількість виміряних ступенів свободи. Не унікальність оновлених моделей - важлива проблема при виявленні пошкоджень, а також при оновленні моделі. Використовуючи метод оновлення моделі, було виявлено, що зрушення однієї частоти не може сам по собі розрізняти зміни в стані підшипника, модуля деформації або розтріскування.

1.1.10 Методи на основі FRF

В деякій літературі основна увага приділяється використанню вимірювань FRF безпосередньо, на відміну від модальних даних, витягнутих з вимірів FRF. Відомі дві основні переваги використання даних FRF. По-перше, модальні дані можуть бути забруднені модальними помилками вилучення на додаток до помилок вимірювання, тому що вони є похідними наборами даних. По-друге,

повний набір модальних даних не можна виміряти в усіх, крім найпростіших структур. Дані FRF можуть надати набагато більше інформації про пошкодження в бажаному діапазоні частот в порівнянні з модальними даними, які витягуються з дуже обмеженого діапазону навколо резонансів. Алгоритм ідентифікації модальних параметрів в реальному часі був застосований для виявлення пошкоджень в експериментальній космічної структури. Зміни постійно відслідковуємої амплітуди, загасання і частот FRF інтерпретувалися як ознаки пошкодження. Перевага безперервного моніторингу в реальному часі полягає в тому, що він дає оператору раннє попередження, так що відповідні дії можуть бути зроблені до того, як станеться катастрофічний відмову. Неточності у виявленні пошкоджень були пов'язані з неточним моделюванням елементів з'єднання і прорізів пазів.

1.1.11 Методи вейвлет-перетворення

Вейвлет-перетворення засновані на ідеї, що будь-який сигнал можна розбити на серію локальних базисних функцій, які називаються вейвлетами. Будь-яка конкретна локальна характеристика сигналу може бути проаналізована на основі масштабних і трансляційних характеристик вейвлетов. Перетворення може застосовуватися і відображатися в просторової або тимчасової області структури. Це відрізняється від перетворення Фур'є, яке зазвичай використовується для відображення з тимчасової області в частотну. Вейвлет-перетворення чутливі до особливостей сигналу, таким як крок. Таким чином, їх можна використовувати для виявлення різкої зміни форми коливань, часто вказує на пошкодження, або для визначення раптової зміни реакції на реакцію часу прискорення. Що стосується космічної галузі важливим питанням при використанні вейвлет-аналізу є кількість виміряних ступенів свободи. Чим вище дозвіл вимірювань в космічній галузі, тим більше інформації може надати вейвлет-аналіз.

1.1.12 Методи з використанням нейронної мережі

Нейронні мережі виникли в результаті вивчення біологічних нейронів і відносяться до обчислювальної структурі, що складається з блоків обробки, що представляють нейрони. Всі нейрони мають кілька входів і один вихід. Нейронні мережі успішно застосовувалися в багатьох різних додатках, включаючи ідентифікацію пошкоджень на основі вібрації Загалом, нейронні мережі особливо застосовні до проблем, тут є значна база даних інформації, але важко вказати явний алгоритм. Нейрона мережа виявилася ефективною, за винятком того, що топологія мережі виявилася критично важливою для продуктивності. У TDNN як вихідний сигнал, так і сигнал через певні проміжки часу подаються на вхід. Обчислювальний час, необхідне для навчання TDNN, було більше, але продуктивність виявилася в цілому краще. Були навчені три мережі, і їх вихідні дані були об'єднані, щоб дати кращі прогнози, ніж за трьома мереж окремо. Кожна мережа була навчена з різними даними, а саме з FRF, модальними даними і даними вейвлет-перетворення. Було висловлено припущення, що підвищення продуктивності було пов'язано з різними структурними змінами, що мають різні відносні видимі ефекти в частотної, модальної і тимчасової областях. Наприклад, зміна демпфірування через додавання губки справила найбільший вплив в тимчасовій області, тоді як FRF показали найбільшу чутливість до просвердлені отвори.

1.1.13 Методи з використанням генетичного алгоритму

Генетичні алгоритми - це методи оптимізації функцій, засновані на випадковій зміні і виборі сукупності рішень. Вони є частиною того, що можна назвати еволюційними алгоритмами, які розроблялися з 1950-х років. Їх перевага перед традиційними алгоритмами оптимізації при підйомі в гору полягає в тому, що вони здатні вирішувати многомодальні топології рішень, які є типовими для задач ідентифікації пошкоджень.

RFV використовується для визначення місця розташування спочатку, а потім на другому етапі генетичні алгоритми використовується для кількісної оцінки пошкодження в успішно ідентифікованих елементах. Метод був продемонстрований на модельованої конструкції з 13 елементів з пошкодженням до 3 елементів.

Для подолання проблеми множинних піків цільової функції використовувався генетичний алгоритм.

1.1.14 Статистичні методи

Для поліпшення рівня техніки в виявленні пошкоджень на основі вібрації необхідні розробки немодельний методів розпізнавання образів, щоб доповнити існуючі методи, засновані на моделях. Ця концепція ідентифікації, яка не

базується на моделях, викликала інтерес до використання виявлення новинок для моніторингу стану. Виявлення новизни пов'язано з ідентифікацією будь-яких відхилень у виміряних даних щодо даних, виміряних в нормальних умовах експлуатації. Характеристики, отримані на основі вимірів, узятих з конструкції в її непошкодженому стані, будуть мати розподіл з відповідним середнім і відхиленням. Якщо конструкція пошкоджена, то може відмітити зміну середнього, дисперсії або того й іншого. Статистичний контроль процесу забезпечує основу для моніторингу розподілу функцій і виявлення нових даних, які не відповідають минулим - «аналіз викидів». Якщо всі інші змінні можна виключити, то зміна характеристик розподілу ознак вкаже на пошкодження. Важливо відзначити, що виявлення пошкоджень, а не їх місцезнаходження і кількісна оцінка, є метою використання статистичного розпізнавання образів. Для створення функцій моніторингу автори використовували функції авторегресії, адаптовані до історичних даних відгуку неушкодженого стану. Середнє значення і дисперсія залишків моделі авторегресії використовувалися для формування діаграм статистичного контролю процесу. Потім та ж авторегресійна модель була адаптована до наступних часових параметрів пошкоджених станів. Отримані залишки були нанесені на контрольні діаграми, і ті точки, які лежали за межами контрольних меж, були підраховані як викиди і оброблені, щоб вказати на зміну в системі. Першим використаним методом був код ланцюжка сигналів (WCC), який характеризує форми сигналів по їх відносного нахилу і кривизні. Відмінності в ухилі і кривизні були оброблені для використання в якості елементів. Другим методом було адаптивне зіставлення шаблонів (ATM), при якому виконувалася покрокова перевірка величини для виявлення відмінностей між двома FRF. Особливість, витягнуті з цього методу був допуск необхідний, щоб тримати сигнал в межах допуску простору опорного сигналу. Другий використовується метод був критерієм гарантії FRF (SAC) і був аналогічний за формулюванням MAC. Максимальне значення 1 вказує на те, що сигнали були ідентичні, в той час як мінімальне значення 0 вказує, що сигнали були абсолютно різними. Третій використаний метод був названий системою еквівалентного рівня деградації (ELODS). Цей метод був заснований на створенні перетворювача, який брав на вхід досліджуваний сигнал FRF і повертав на виході той же самий сигнал, якщо структура не була пошкоджена, але повертав перекручену версію вхідного сигналу, якщо структура була пошкоджена. Це забезпечило функцію ідентифікації спотворення як ознака ідентифікації. З змодельованими даними за відсутності шуму рейтинг ефективності методів був наступним: ELODS, WCC, ATM, а найменш ефективним був SAC. З даними з автомобільного моста тільки метод WCC зміг виявити тріщину. Всі методи дозволяють виявити тільки наявність, а не місце або ступінь пошкодження.

Методи статистичного розпізнавання образів, які базуються на моделях, не вимагають створення корелятивною числовий моделі, що є трудомісткою і складним завданням. Вони також підходять для наборів даних, отриманих тільки при зовнішньому порушення, наприклад, при русі або вітрового навантаження

на конструкцію мосту. Недоліком цих методів є те, що вони, ймовірно, обмежені ідентифікацією Рівня 1 або, можливо, Рівня 2. недавно запропонувала метод виявлення і локалізації ушкоджень, заснований на статистичній моделі, що використовує залишок на основі підпростору і статистичний аналіз сукупної чутливості залишку до пошкодження.

Збиток позначається, коли значення залишкової суми проходить статистично обґрунтоване правило прийняття рішення. Локалізація ушкоджень визначається за допомогою чутливості залишку до пошкодження в елементах конструкції, розрахованої за допомогою аналітичної моделі. Метод застосуємо до випадків, коли доступні тільки вихідні дані, і був розроблений на основі методів стохастичною ідентифікації на основі підпросторів.

1.2 Класифікація стану обладнання за допомогою Dictionary Learning

Процеси машинного навчання і профілактичного обслуговування в деякій мірі схожі. В обох випадках є етап збору даних, етап вилучення ознак і етап класифікації. Для профілактичного обслуговування можуть бути корисні нові методи обробки даних, і щоб зробити методи машинного навчання життєздатною альтернативою, методи необхідно адаптувати до потреб прогнозного обслуговування. Основна проблема в профілактичному обслуговуванні - якомога раніше передбачити режими відмови, щоб можна було вжити заходів для продовження терміну служби відслідковується системи. Крім того, методи повинні враховувати нетипові умови, які є звичайними для сигналів підшипників, такі як імпульсна модуляція сигналу через механічну нещільності. Ідентифікація фізичних характеристик, що впливають на сигнал, важлива для досягнення більш широкого застосування підходу машинного навчання, оскільки виявлення відхилень щодо нормального критерію може бути недостатньо, коли в сигналі переважають нетипові умови. В даний час використання методів машинного навчання в дослідженнях моніторингу стану обмежується доступністю загальнодоступних наборів даних, необхідних для навчання. Отримання даних для навчання є важкою справою через великої кількості потенційних умов і комбінацій підшипників. Таким чином, більшість досліджень машинного навчання в цій області, як правило, зосереджена на виявленні локалізованих і помічених дефектів, а не на наданні інформації про атрибути пошкоджень. Етап вилучення ознак є загальним для областей прогнозного обслуговування і машинного навчання. Таким чином, потрібна гнучкий і ефективний підхід до вилучення ознак.

Успіх розрідженого кодування з навчанням за словником в широкому спектрі областей застосування мотивував роботу також в області моніторингу стану. Використовують попередньо сконструйований словник на основі атомів Габора для розкладання сигналу з використанням алгоритму пошуку збігу. Отримане розкладання використовується для вилучення сигнатури вібрації підшипника, яка потім порівнюється з характеристичними частотами дефектів підшипника для ідентифікації несправності. Однак тільки в 2011 році стали першими, хто використав розріджений кодування з навчанням за словником для класифікації несправностей підшипників. Ці автори використовують інваріантні до зсуву словників, вивчені незалежно від різних станів несправностей підшипників, в якості вхідних даних для багатокласового лінійного дискримінантного аналізу для класифікації розмірів і розташування несправностей. Після дослідження кілька дослідників розглянули можливість використання словникового навчання для виявлення несправностей і класифікації сигналів від обертових машин, таких як підшипники кочення або редуктори. Результуючі імпульсні функції порівнюються з характеристичними частотами для визначення місця несправності. Отримані приховані компоненти використовуються для виявлення несправностей в підшипнику або коробці передач шляхом перетворення прихованих компонентів в частотну область.

використовують приховану марковську модель для класифікації несправностей. Спочатку проводиться дискретизація сигналів вібрації з використанням вейвлетного перетворення в різних масштабах. Потім вивчення словника застосовується до спектрів різних класів. Отримана помилка реконструкції розрідженого уявлення використовується в поєднанні з теорією відомств для визначення виникнення і серйозності несправностей. Вони використовують алгоритм навчання словника K-SVD для вивчення матриці ознак. Розмірність матриці ознак додатково зменшується з допомогою PCA. Згодом алгоритм K найближчих сусідів використовується для класифікації різних станів несправності. Більшість цих досліджень з вивчення словника для виявлення несправностей за допомогою сигналів підшипників зосереджені на використанні вивчених атомів для класифікації несправностей і на те, як такі класифікатори можуть бути покращений. Крім того, в цих дослідженнях використовуються дані контрольованих експериментів, в яких розломи штучно викликані або відомі раніше. Інших повідомлень про застосування словникового навчання до сигналів акустичної емісії в контексті моніторингу стану не надходило. Вивчені функції, засновані на вивченні словника, використовувалися в декількох дослідженнях, присвячених класифікації. Однак використання схеми класифікації для вирішення проблеми виявлення несправностей в додатках моніторингу стану стикається з низкою проблем. Наприклад, як часто слід перенавчати класифікатор, щоб врахувати природний знос обертається машини? Якщо ці оновлення виконуються дуже часто, розвиток несправностей, на розвиток яких йде багато часу, може бути розцінена як справний стан машини. Альтернативний підхід полягає в тому, щоб зробити вибір функцій і виявлення аномальних умов онлайн-процесом шляхом розробки індикаторів, що підходять для онлайн-вивчення функцій. Індикатор стану - це кількісна міра продуктивності або робочого стану машини. Точно так же функція - це вимірна характеристика, яка використовується для моделювання сигналу. Як правило, традиційні підходи і підходи машинного навчання до моніторингу стану використовують індикатори стану в тимчасовій і частотній областях, такі як RMS і зміст енергії в певних частотних діапазонах. Однак такі індикатори визначаються фахівцями-людьми, а параметри зазвичай настроюються вручну для кожної програми. Пропонований тут підхід призначений для доповнення і потенційної заміни звичайних індикаторів стану індикаторами, отриманими в процесі неконтрольованого навчання властивостями.

Двома ключовими аспектами методів машинного навчання є адаптація і узагальнення. Адаптація вимагає, щоб система, яка реалізує процес машинного навчання, могла адаптуватися для підвищення його точності. Це дозволяє поліпшити за рахунок повторення. Узагальнення - це застосування того, що система вже вивчила, до даних зі зв'язаними характеристиками. Ці дві концепції можна об'єднати в цикл, і з великою кількістю даних система може зробити більш точним аналіз нових даних. Одним з підходів до машинного навчання є метод описового або неконтрольованого навчання, мета якого полягає в тому, щоб ідентифікувати шаблони у вхідних або вхідних-вихідних даних, які

задовольняють конкретним обмеження. Багато методів і алгоритмів машинного навчання були натхненні біологічними або еволюційними процесами. Одним з мотивів використання такого натхнення є спроба вирішити проблему невизначеності. Коли цей процес застосовується до проблеми виявлення несправностей в підшипниках кочення, сигналом можуть бути, наприклад, дані про вібрації або акустичної емісії, а вихідні дані будуть включати інформацію про стан підшипника і машини. Етап виділення ознак уточнює структуру даних, а етапи моделі і класифікації визначають закономірності в наборі ознак, щоб їх можна було згрупувати разом і класифікувати . Робота, представлена в цій дисертації, зосереджена на стадії вилучення ознак. Зокрема, про те, як вивчення функцій може бути використано для виявлення аномалій і змін стану обертається машини.

У деяких випадках виходом є місце несправності (наприклад, внутрішнє кільце, зовнішнє кільце або елемент кочення). В інших випадках висновок являє собою просто класифікацію наявності несправності. Однак в більшості досліджень в цій області основна увага приділяється виявленню локальних дефектів підшипників в контрольованих експериментах.

1.3 Виявлення проблем за допомогою виявлення викидів

Викид - це спостереження або набір спостережень, які значно відхиляються від того, що вважається нормальним. Визначення нормальної поведінки пов'язано з тим, як розподіляються дані, що зазвичай визначається на основі історичних даних. Детектор викидів може порівнювати нові спостереження з загальним розподілом моделі або з розподілом підмножини найостанніших спостережень. Якщо різниця між новим спостереженням і моделлю значна, то детектор буде вважати це спостереження викидом. До проблеми виявлення викидів можна підійти кількома способами. Один з підходів - моделювати тільки нормальну поведінку. Інший підхід - моделювати як нормальне, так і ненормальну поведінку.

1.4 Оцінка стану за допомогою виявлення змін та типи аномалій

Виявлення змін - це проблема виявлення змін розподілу ймовірностей випадкової величини або тимчасового ряду. Як правило, нові спостереження слідують в послідовному порядку в потоках даних часових рядів. Точка зміни - це перехід між різними розподілами даних. Така зміна даних може бути пов'язана з відхиленням від норми або нормальними змінами які раніше не спостерігалися в системі, яка виробляє дані. Основна характеристика цього типу виявлення полягає в тому, що зміна є постійною і не обмежується одним викидом або набором викидів. Приклади різних типів аномалій представлено на рисунку 1.1:

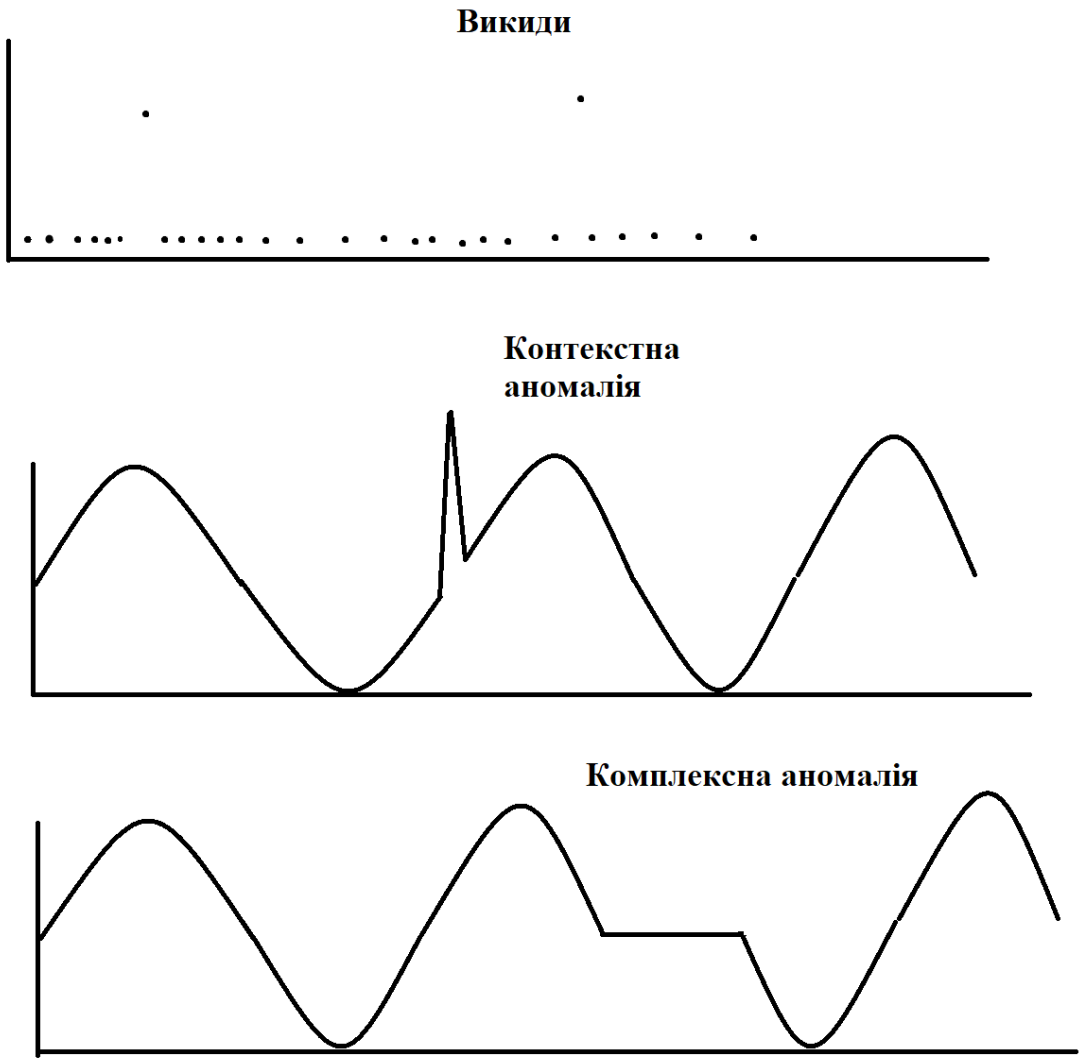

Рисунок 1.1 - Приклади різних типів аномалій

Аномалія цього типу також відома як умовна аномалія і відноситься до вимірювання, яке є аномальним в певному контексті, але не в інших. Контекст визначається загальною або локальною структурою даних. В цьому випадку окремий вимір може вважатися нормальним при певних умовах, але аномальним в інших умовах.

Колективна аномалія виникає, коли група вимірювань аномальна по відношенню до нормального розподілу даних. Індивідуальне вимір саме по собі не є колективною аномалією. Однак, коли окремий вимір розглядається з групою вимірювань, набір вимірювань може бути колективної аномалією.

Зазвичай механічна система, така як підшипник, схильна до контекстних або колективним аномалій, враховуючи характер і тимчасової аспект даних, доступних від датчиків, які контролюють ці системи. Система, що працює довгий час, зазнає змін через природній знос системи. Такий знос відноситься до зміни, яке вважається нормальною поведінкою. Таким чином, в ідеалі метод виявлення аномалій повинен бути здатний враховувати такі зміни і адаптуватися до них, оскільки вони не відповідають аномалій в системі. При розгляді виявлення аномалій в потоці даних виникають дві основні проблеми. Одна з

проблем пов'язана з тим, як дані збираються і обробляються. В автономному сценарії дані збираються і обробляються партіями. Навпаки, онлайн-обробка даних вимагає послідовної роботи в реальному часі. Друга проблема пов'язана з обмеженими ресурсами, доступними для виявлення аномалій. У разі потокової передачі даних постійно з'являються нові дані. Таким чином, дані необхідно якось обробляти і агрегувати, що ускладнює завдання аналізу і обробки.

Неконтрольоване навчання функцій - це підхід машинного навчання, при якому алгоритми використовуються для вивчення представлення функції з немаркованих даних. Таким чином, це корисно для моделювання складних даних, для яких важко отримати попередні знання. Мета - вивчити уявлення, яке можна використовувати для вирішення інших завдань. Сенс цього підходу полягає в тому, що вивчені функції забезпечать краще уявлення, ніж необроблений сигнал. По-перше, немарковані дані обробляються неконтрольованим алгоритмом навчання функцій для створення набору вивчених функцій, які відповідають ідентифікованої структурі. Вивчені функції використовуються кодувальником, який витягує вектор ознак з нових даних, які, наприклад, можуть використовуватися для прогнозування різних умов.

Автокодіровщік - це неконтрольована нейронна мережа, яка використовується для вилучення ознак і зменшення розмірності, які зазвичай використовуються в глибокому навчанні і характеризуються тим же кількістю входів, що і виходи. Автокодіровщікі - це ANNs з прямим зв'язком, які навчені передбачати введ. Ідея полягає в тому, що один або кілька прихованих шарів мають меншу розмірність, ніж видимі шари, так що інформація, необхідна для відновлення введення, стискається в прихованих шарах. Подальші розробки базової моделі і концепції автокодіровщіка включають розріджену регуляризацію активації прихованих блоків, шумоподавляющіе автокодіровщікі і складові автокодіровщікі, в яких кілька моделей накладаються один на одного.

За допомогою складеного автокодіровщіка вони вивчають нелінійні проекції спектрів сигналу, які використовуються для точного налаштування нейронної мережі, що використовується для розрізнення різних зазначених станів здоров'я.

Методи кластеризації використовуються для ідентифікації підмножин даних зі схожими атрибутами. Таким чином, мета полягає в тому, щоб розділити набір з n об'єктів на K груп таким чином, щоб схожість між об'єктами в кожній групі було високим, а подібність між об'єктами в різних групах була низькою. Алгоритм K-середніх заснований на евклідових відстанях між точками даних і середньому для кожного кластера. Шляхом мінімізації квадрата відстані по всім кластерам найближчі точки даних призначаються одному і тому ж кластеру. Інші алгоритми кластеризації включають суміш алгоритму Гауса і варіантів алгоритму K-means.

Алгоритм K-means відносно простий в реалізації. Таким чином, різні проблеми були вирішені з використанням K-means, включаючи виявлення несправностей. Реалізація являє собою двоетапну процедуру, де на першому етапі визначається, чи існує несправність, а на другому етапі визначається

місцезнаходження несправності. Розглядаються алгоритми K-means, ієрархічна кластеризація і максимізація очікування. Метою дослідження є розробка правила для визначення кількості кластерів при роботі з погано розділеними даними.

Розріджене кодування можна комбінувати з алгоритмами неконтрольованого навчання ознак для одержання стисненого представлення сигналу. При розрідженому кодуванні вхідний сигнал зазвичай представляється як лінійна комбінація характеристик / функцій. Набір функцій іноді називають словником, і, як правило, кількість функцій більше, ніж розмірність вхідного сигналу, що призводить до переповнення уявлення. Словник, який використовується для розрідженого кодування, може бути або визначеним і постійним, або його можна вивчити. Вейвлети - це один з добре відомих типів функцій, які використовуються для побудови словників констант.

Методи аналізу в частотної області більш популярні для аналізу сигналів вібрації, тому що вони пропонують можливість звернення до періодичних подій, викликаним дефектами гонки. Наприклад, локальні дефекти в підшипнику викликають дефекти певної частоти. Ці частоти дефектів можна розрахувати за такими формулами.

Частота проходу кулі по зовнішньої обоймі (BPFO):

$$f(Hz) = \frac{nf_r}{2}\{1 - \frac{d}{D}\cos\emptyset\} \qquad (1.1)$$

Частота проходу кулі по внутрішній обоймі (BPFI):

$$f(Hz) = \frac{nf_r}{2}\{1 + \frac{d}{D}\cos\emptyset\} \qquad (1.2)$$

Основна частота сепаратора (FTF):

$$f(Hz) = \frac{f_r}{2}\{1 - \frac{d}{D}\cos\emptyset\} \qquad (1.3)$$

Частота обертання кульки (ролика) (BSF):

$$f(Hz) = \frac{D}{2d}\{1 - (\frac{d}{D}\cos\emptyset)^2\} \qquad (1.4)$$

Тут n - кількість тіл кочення, а fr - частота обертання валу. Див. Визначення D, d і φ на рисунку 1.2:

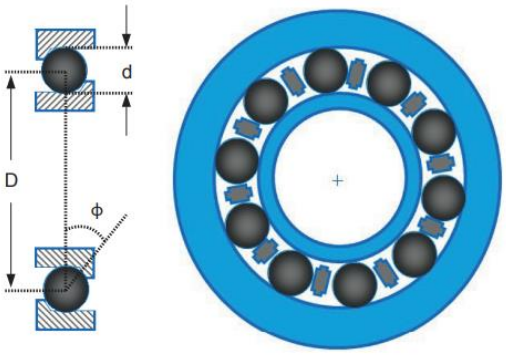

Рисунок 1.2 - Графічний опис змінних, використовуваних для розрахунку частот характерних дефектів підшипників.

Використання методів частотної області додатково полегшується за рахунок швидкого перетворення Фур'є (ШПФ), яке дозволяє ефективно отримувати частотний спектр. Простий метод аналізу полягає в прямому порівнянні спектра сигналу з характеристичними частотами дефектів підшипників. Приклади відображення спектрів представлено на рисунках 1.3 та 1.4:

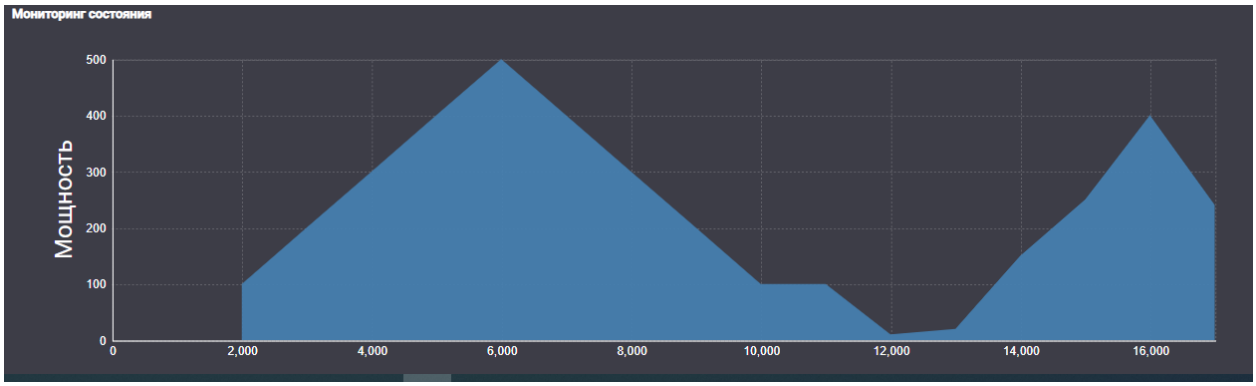

Рисунок 1.3 – Відображення спектру

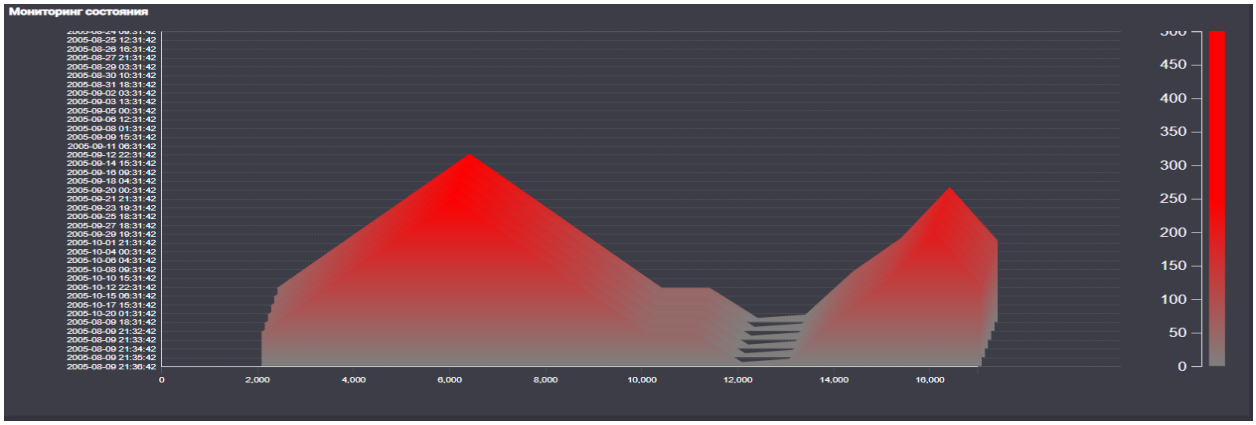

Рисунок 1.4 – Відображення спектру з історичними даними

БПФ - ефективний алгоритм аналізу даних з дискретним часом. Процес перетворює тимчасову запис в дискретну спектральну запис, тут кожна вибірка представляє собою дискретний частотний сегмент зони Найквиста. Загальна кількість вихідних відділків дорівнює кількості відділків в вихідної записи часу, яке в більшості випадків являє собою число в біноміальними ряду. Спектральні дані містять інформацію про амплітуду і фазі, яка може бути представлена в прямокутної або полярної формі. У прямокутній формі одна половина вікон БПФ містить інформацію про величину, а інша половина містить інформацію про фазу. В полярній формі одна половина вікон БПФ містить реальний результат, а інша половина - уявний результат. У деяких випадках корисна інформація як про величину, так і про фазу, але співвідношення величина / частота часто містить достатньо інформації для виявлення ключових змін. Для пристроїв, які пропонують тільки результати за величиною, кількість вікон БПФ дорівнює половині відділків в вихідної записи в тимчасовій області. Ширина комірки БПФ дорівнює частоті дискретизації, поділеній на загальну кількість записів. У певному сенсі, кожен елемент БПФ схожий на окремий смуговий фільтр в тимчасовій області. На малюнку 5 представлений приклад реального датчика вібрації MEMS, який виробляє вибірку зі швидкістю 20480 вибірок в секунду (SPS) і починає з записів по 512 точок. В цьому випадку датчик надає тільки інформацію про величину, тому загальна кількість осередків становить 256, а ширина осередку дорівнює 40 Гц (20480/512). Приклад значень реального датчику представлено на рисунку 5:

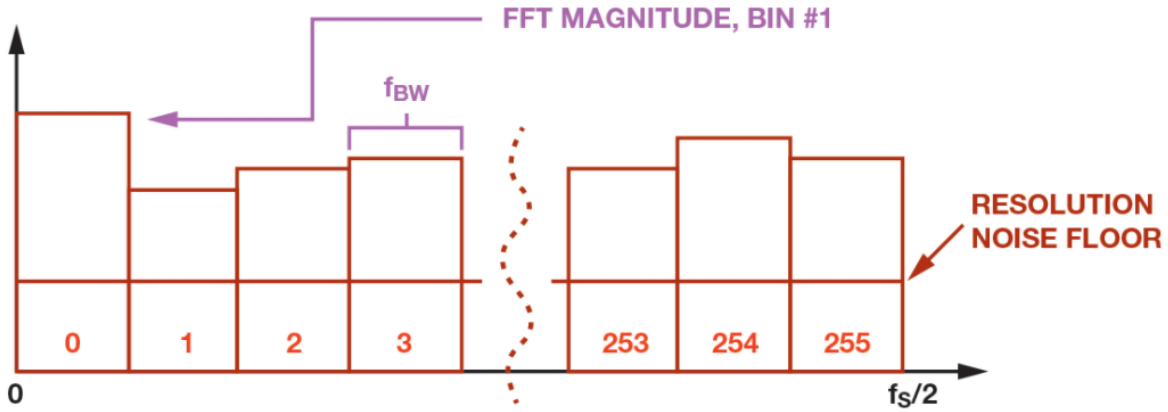

Рисунок 1.5 - приклад реального датчика вібрації MEMS

Загальний шум (середньоквадратичне значення) дорівнює добутку щільності шуму (~ 240 мкг / √Гц) і квадратного кореня з ширини інтервалу (√40 Гц), або ~ 1,5 мг середньоквадратичного значення. Для низькочастотних додатків, де шум має тенденцію мати найбільший вплив на дозвіл вібрації, фільтр перед процесом БПФ може допомогти поліпшити дозвіл по частоті і величиною, не вимагаючи зміни частоти дискретизації АЦП. Зменшення частоти дискретизації 20480 SPS в 256 разів збільшує дозвіл по частоті в 256 разів при одночасному зниженні шуму в 16 разів.

Одним з ключових переваг використання БПФ є те, що він дозволяє просто застосовувати спектральні сигнали тривоги. На малюнку 6 представлений приклад, який включає п'ять незалежних спектральних сигналів тривоги, які контролюють власну частоту в машині (№1), її гармоніки (№2, №3 і №4) і широкосмуговий контент (№5). Попереджувальний і критичний рівні відповідають рівням в профілі залежності стану машини від часу. Частоти запуску і зупинки завершують визначення змінної процесу, представлене цим співвідношенням. При використанні вбудованого процесора змінні визначення спектральних сигналів тривоги (частоти запуску / зупинки, рівні попереджень / критичних сигналів тривоги) можуть перебувати в настроюються осередках регістрів, які використовують цифрові коди для конфігурації. Використання тих же масштабних коефіцієнтів і схеми нумерації осередків може значно спростити цей процес. Приклад реалізації спектральних сигналів тривоги представлено на рисунку 1.6:

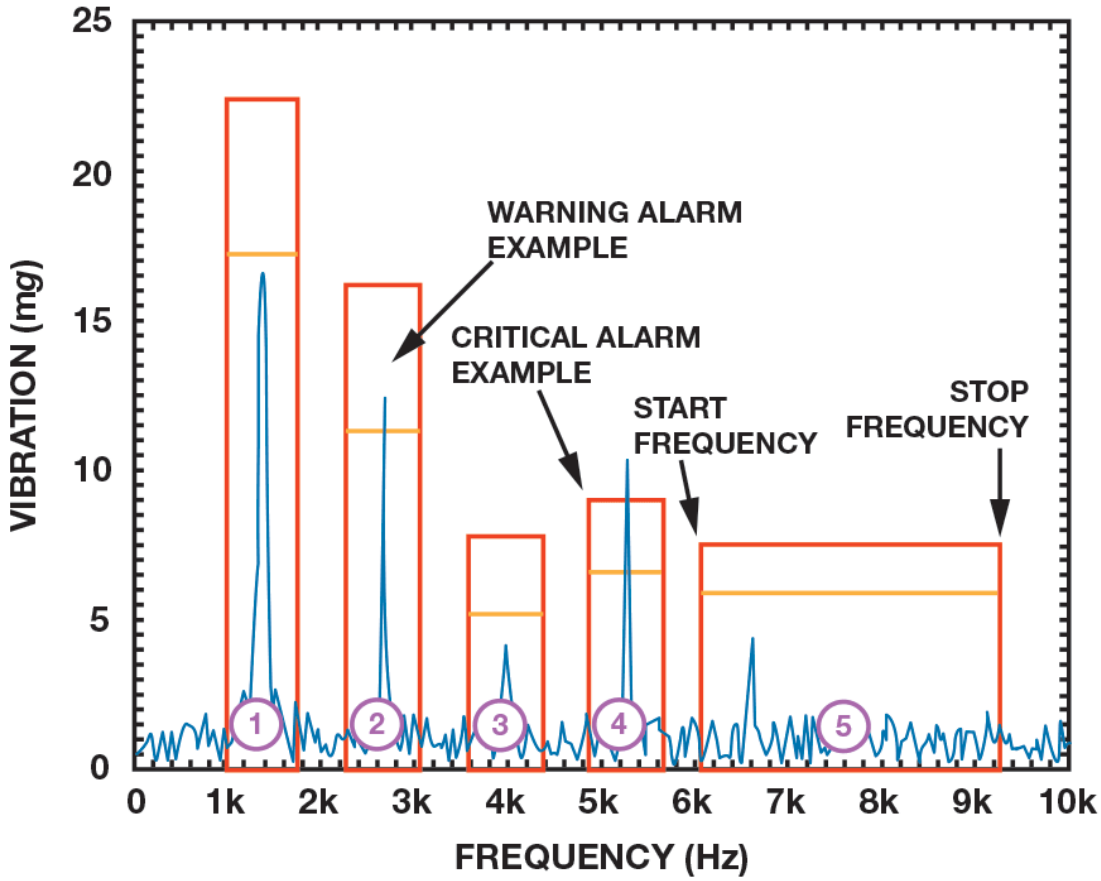

Рисунок 1.6 – Приклад спектральних сигналів тривоги

1.5 Висновки за першим розділом

Проведено аналіз методів оцінки стану обладнання та наявних технічних рішень діагностики обладнання. У результаті для використання у системі був обраний метод заснований на Dictionary Learning. Сформульовано завдання для розробки системи вібраційної діагностики.

2 ПЛАНУВАННЯ ЕКСПЕРИМЕНТУ ДЛЯ ВІБРАЦІЙНОЇ ДІАГНОСТИКИ

2.1 Серверна частина системи вібраційної діагностики

Реалізація серверної частини представлена на рисунку 2.1.

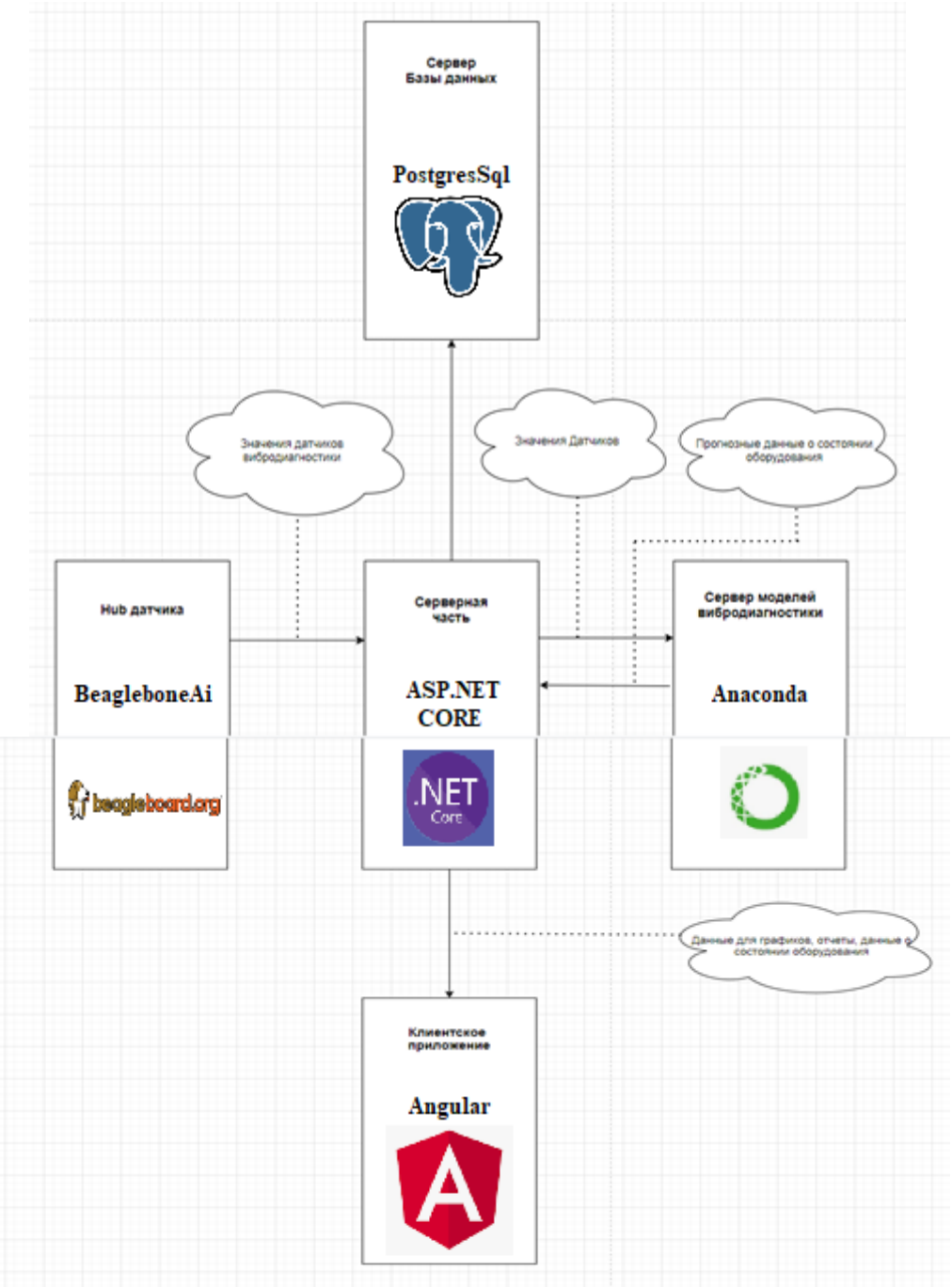

Рисунок 2.1 – Архітектура системи вібраційної діагностики

2.2 Апаратна частина системи вібраційної діагностики

Датчик базується на мікроконтролері STM32L476 з ядром ARM Cortex M4. Цей мікроконтролер включає вбудований високоточний генератор опорної частоти і співпроцесором прискорення математичних операцій з плаваючою крапкою (FPU). Bluetooth реалізований за допомогою Texas Instrument CC2650 в версії MODA. Для отримання вібраційних даних використовується MEMS датчик. Реалізована плата показана на рисунку 2.2.

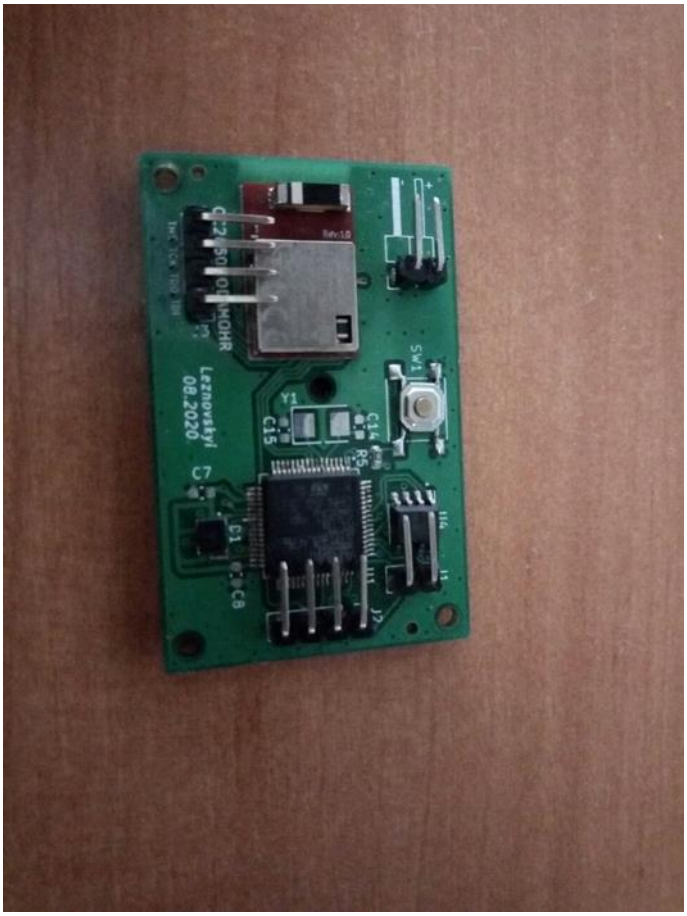

Рисунок 2.2 – Підсумковий варіант плати

Технічні характеристики:

1) смуга пропускання: 6kHz;
2) частота дискретизації: 25.6kHz;
3) роздільна здатність: 16 біт ;
4) максимальне значення віброприскорення: 2G.

Для обміну даними між MEMS датчиком та мікроконтролером використовується інтерфейс SPI. При розробці ПО для мікроконтролера використовувалася IDE STM32IDE. При розробці плати використовувалося ПО KiCad.

Так як датчики використовують інтерфейс Bluetooth для обміну даними потрібний пристрій який може отримувати пакети даних від датчику за допомогою Bluetooth та відправляти ці данні за допомогою мережі Internet на віддалений сервер. Цей пристрій також може використовуватися для проміжних обчислень. За основу цього пристрою був взятий міні комп’ютер з відкритою документацією beaglebone ai. Ви можете побачити міні комп’ютер beaglebone ai на рисунку 2.3.

Рисунок 2.3 – міні комп’ютер beaglebone ai

Вибір даного міні комп’ютера зумовлений тим що цей проект є відкритим. Це дозволить самому при необхідності дороблятиа та виробляти цей міні комп’ютер. Також даний міні комп’ютер розрахований на обробку сигналів за допомогою високоефективних DSP(Digital Signal Processor) та прискорювачів нейронних мереж. Робота з датчиками вібраційної діагностики проходить за допомогою Bluetooth модулю який також має даний міні комп’ютер . Для доступу у мережі Інтернет може бути використаний WiFi та Internet.

Beaglebone ai працює під управлінням Linux а саме Debian. Для реалізації функціональності роботи з вібраційним датчиком була реалізована програма на

мові програмування C++. Програма реалізує роботу з BLE, має функції буферізації та передає отримані показники на віддалений сервер. Якщо зв'язок з віддаленим сервером відсутній то програма накопичує отриманні вібраційні показники у пам'яті та при відновленні зв'язку передає їх разом з міткою часу. Схему взаємодії хаба можливо побачити на рисунку 2.4.

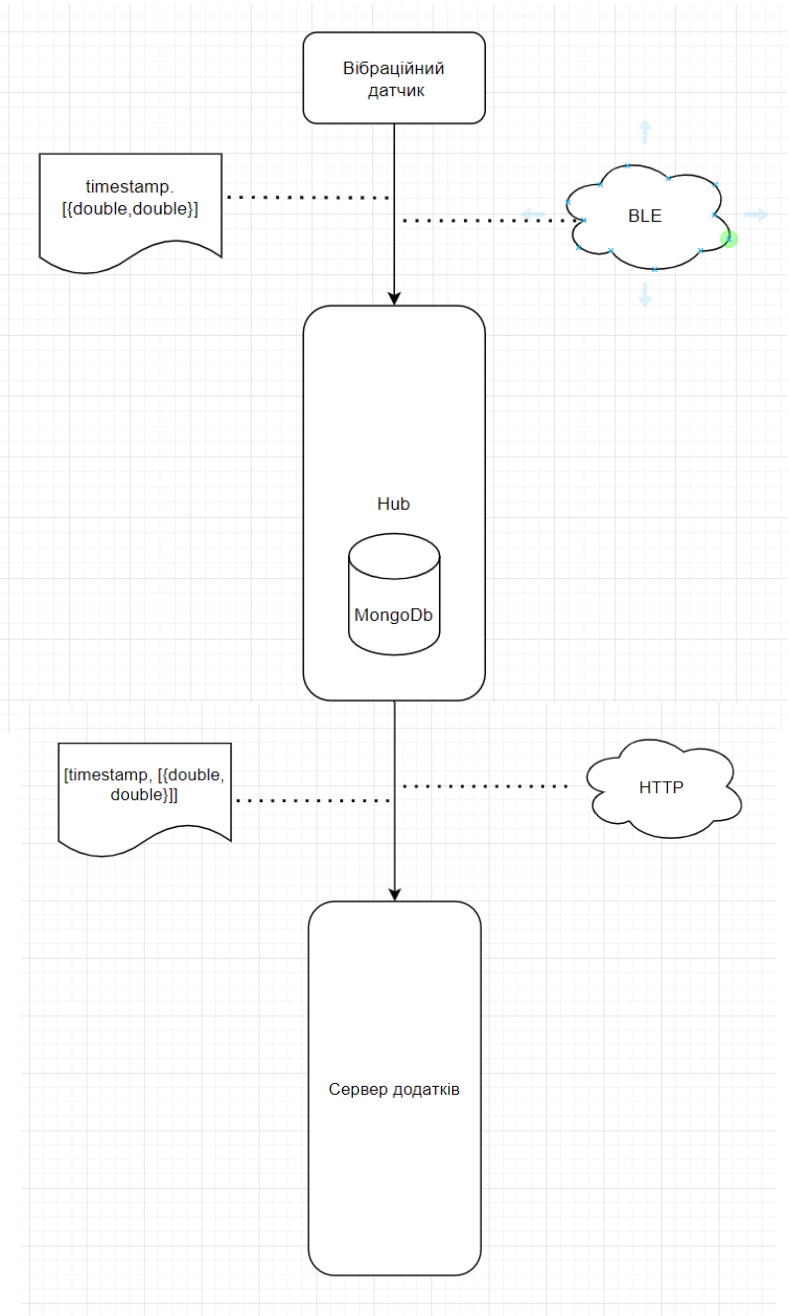

Рисунок 2.4 – Схема взаємодії датчиків з хабом

При передачі між датчиком на хабом використовується пакет наступного вигляду – {timestamp, [{double, double}] } цей пакет містить мітку часу та дані у частотній області які представлені за допомогою масиву елементів з полями double , double в яких зберігається частота на вібраційне прискорення.

Так як датчик вібрації сам переводить показники з часової області у частотну за допомогою швидкого перетворювання Фур'є ці пакети не займають

багато місця та з легкістю можуть бути передані за допомогою BLE на значні відстані з незначними витратами заряду батареї. Якщо будуть потрібні данні у часовій області (наприклад для аналізу) то можливо відновити данні за допомогою оборотного перетворення Фур'є.

Ці пакети даних хаб передає на сервер додатків, якщо зв'язку з сервером немає то хаб починає накопичувати показники у буфер який реалізований за допомогою NoSQL бази даних MongoDb. У подальшому хаб сам зможе аналізувати ці показники за допомогою прискорювачів нейронних мереж.

Для оновлення забезпечення датчиків також використовується хаб. Хаб посилає запроси щоб перевірити теперішню версію забезпечення. Якщо версія відрізняється від тієї версії яку мають датчики підключенні до хабу то хаб починає закачку нової версії. Коли версія закачана перевіряється контрольна сума. Якщо сума співпадає то хаб починає оновлення програмного забезпечення датчиків. Побачити схему оновлення можливо на рисунку 2.5.

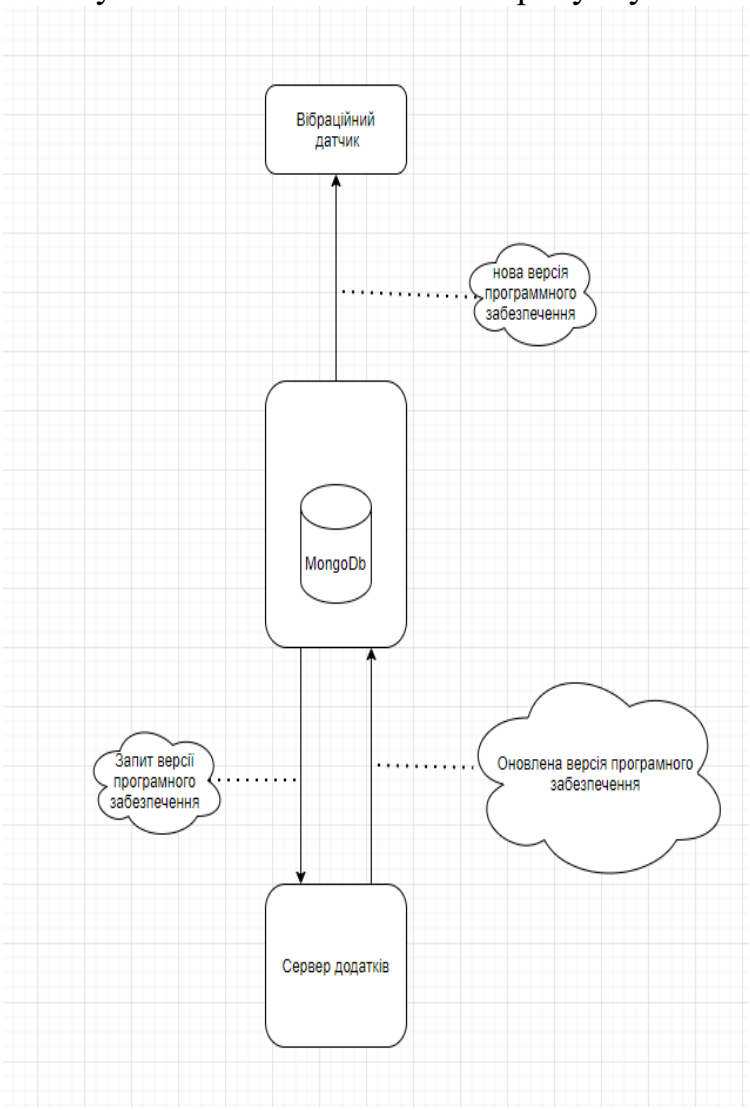

Рисунок 2.5 – схема оновлення ПЗ вібраційного датчика

2.3 Вибір БД для серверної частини системи вібраційної діагностики

Програмне забезпечення для обробки даних з контролерів температури повинне зберігати дані протягом тривалого часу. На сьогоднішній день популярними безкоштовними СУБД є: MS SQL Express, MySQL, PostgreSQL.

MySQL – це СУБД з відкритим кодом. MySQL високо оцінена користувачами. Ця СУБД добре себе поводить з середніми та малими обсягами даних, є документація в інтернеті та можливість онлайн консультування.

PostgreSQL є потужною СУБД з відкритим кодом. PostgreSQL розвивається більш ніж 15 років та має перевірену архітектуру, яка заслужила високу репутацію за надійність, цілісність даних і коректність. PostgreSQL має складні функції, такі як Multi-Version управління паралелізму (MVCC), асинхронну реплікацію, вкладені транзакції (точки збереження).

MS SQL Server – це СУБД від Microsoft яка має безкоштовну редакцію. MS SQL Server дозволяє використовувати Transact-SQL та разом з ним такі технології, як Microsoft ADO.NET Entity Framework та LINQ. MS SQL Server має глибоку інтеграцію з Visual Studio і SQL Server Management Studio. Взаємодіє з багатьма мовами програмування. Має добру масштабованість та надійність.

Для розробки програмного забезпечення для обробки даних з контролерів температури буде використовуватися PostgreSQL що вона надійна, відкрита та прогнозована.

2.4 Проектування бази даних ПЗ

Для ідентифікації користувача та захисту від несанкціонованого доступу були створенні наступні сутності:

1) RoleClaims - Містить додаткові дані для ролей що бути необхідні при роботі;
2) Roles – Містить доступні ролі для користувачів;
3) UserClaims - Містить додаткові дані для користувачів що бути необхідні при роботі;
4) UserLogins – Містить сесії входу користувачів;
5) UserRoles – Містить ролі користувачів;
6) Users – Містить основні данні про користувачів. Наприклад E-mail, номер телефону і т.д;
7) UserToken – Містить засолені паролі користувачів.

Сутності в БД які представляють бізнес – логіку:

1) AverageValue – Містить данні о показникам числових сенсорів. Наприклад сенсор температури;
2) Sensor – Містить данні о сенсорах. Наприклад о ідентифікаторі сенсору, назві;
3) SensorValue – Містить дані о показниках вібраційних сенсорів. Наприклад дата з часом та частотні показники;

4) Thing – Містить дані о обладнанні до якого прив’язуються сенсори. Наприклад назву та список сенсорів;

Визначення зв'язків між сутностями:

1) Сутність «Sensor» пов'язана з сутністю «SensorValue» ставленням «1: N» -у одного сенсору може бути багато значеннь;
2) Сутність «Thing» пов'язана з сутністю «Sensor» ставленням «1: N» -у одного обладнання може бути багато сенсорів;
3) Сутність «NetRoles» пов'язана з сутністю «RoleClaims» ставленням «1: N» -у однієї ролі може бути багато додаткової інформації;
4) Сутність «NetRoles» пов'язана з сутністю «UserRoles» ставленням «1: N» -у однієї ролі може бути багато користувачів;
5) Сутність «Users» пов'язана з сутністю «UserRoles» ставленням «1: N» -у одного користувача може бути багато ролей;
6) Сутність «Users» пов'язана з сутністю «UserLogins» ставленням «1: N» -у одного користувача може бути багато сесій входу;
7) Сутність «Users» пов'язана з сутністю «UserClaims» ставленням «1: N» -у одного користувача може бути багато додаткової інформації;
8) Сутність «Users» пов'язана з сутністю «UserTokens» ставленням «1: N» -у одного користувача може бути багато додаткової інформації.

Розробляється база даних призначена для зберігання інформації про користувачів, сенсорів, подій, значень сенсорів, обладнання:

1) RoleClaims {Id, RoleId};
2) Roles {Id, Name, NormalizedName, ConcurrencyStamp };
3) UserClaims {Id, UserId, ClaimType, ClaimValue};
4) UserLogins{LoginProvider, ProviderKey, ProviderDisplayName, UserId};
5) UserRoles{UserId, RoleId };
6) Users{Id, UserName, NormalizedUserName, Email, NormalizedEmail, EmailConfirmed, PasswordHash, SecurityStamp, ConcurrencyStamp, PhoneStamp, PhoneNumberConfirmed, TwoFactorEnable, LockoutEnd, LockoutEnabled, AccessFailedCount};
7) UserTokens{UserId, LoginProvider, Name , Value};
8) Sensor{Id, Name,, ThingId, Type};
9) SensorsValues {Id, SensorId, FreqValues, Date};
10) Thing {Id, Name};
11) AverageValue {Id, SensorId, Date, Value, State}.

Фізична модель даних та опис таблиць з уточненими зв'язками показана на рисунку 2.6

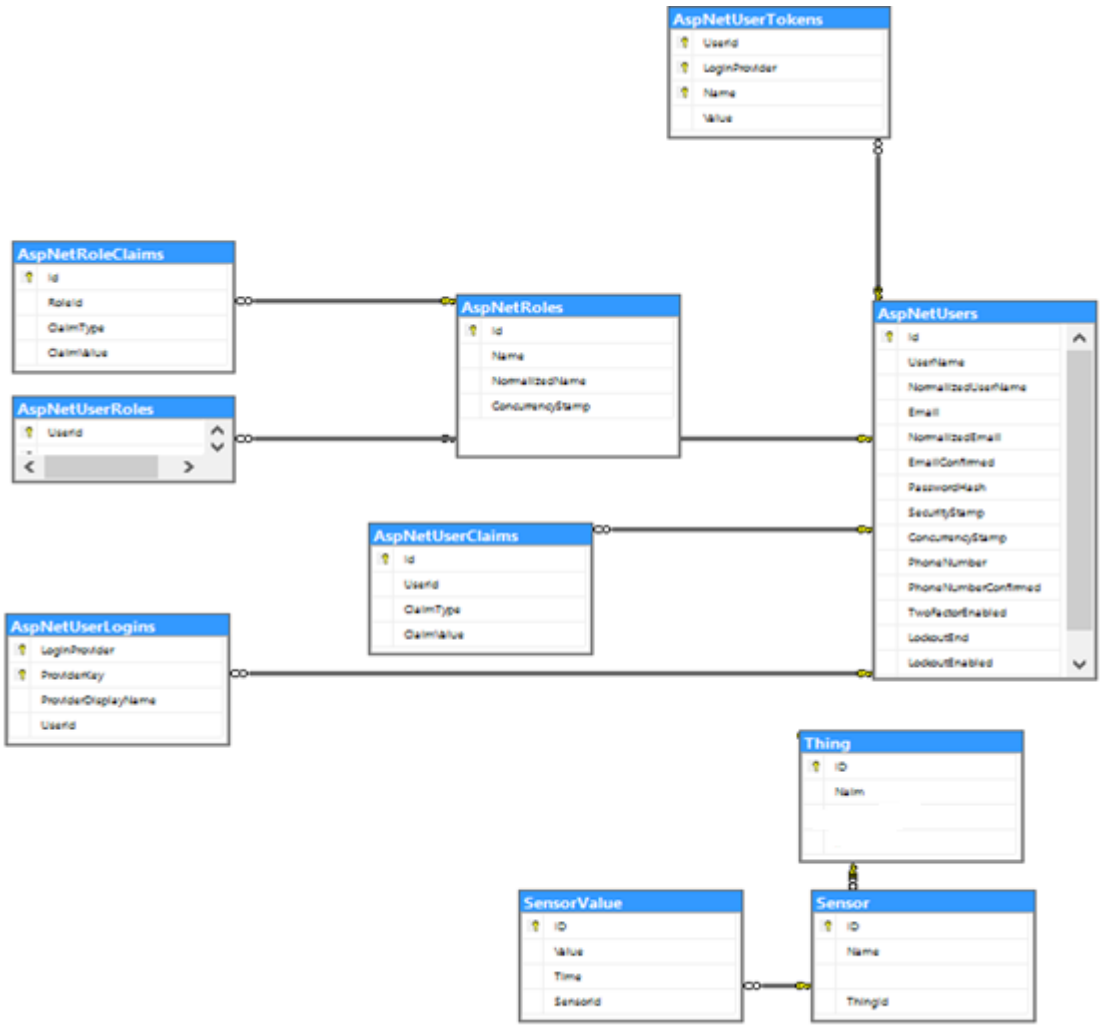

Рисунок 2.6 - Фізична модель даних та опис таблиць

2.5 Розробка серверної частини системи вібраційної діагностики

Однією з головних вимог до розроблюваної системи є забезпечення можливості одночасної роботи та обробки даних для багатьох користувачів.

При проектуванні буде побудована діаграма класів, цей різновид діаграми описує предметну галузь в понятті об’єктно – орієнтованого програмування, вона конкретизує об’єкти та зв’язки між ними. У кожного класу має бути ім’я , поля та методи.

В вибраній архітектурі виділяють три компонента: моделі що описують об’єкти предметної галузі, відображення які відображають інформацію користувачу та контролери які обробляють поступаючи запити. В проекті ці компоненти виділенні в окремі проекти.

Проект IoTMonitoring.Core містить моделі що представляють собою класи моделей для ORM:

1) AverageValue – Значення сенсорів представлені у числових показниках;
2) Sensor – Сенсори обладнання (датчики температури, датчики вібраційної діагностики і тд.);
3) Thing – Обладнання моніторинг якого потрібно проводити;
4) SensorValue – Значення сенсорів вібраційної діагностики;
5) SensorState – Стан сенсору відповідно до поточних показників;
6) SensorType – Тип сенсору.

На рисунку 2.10 можливо побачити взаємозв'язок класів.

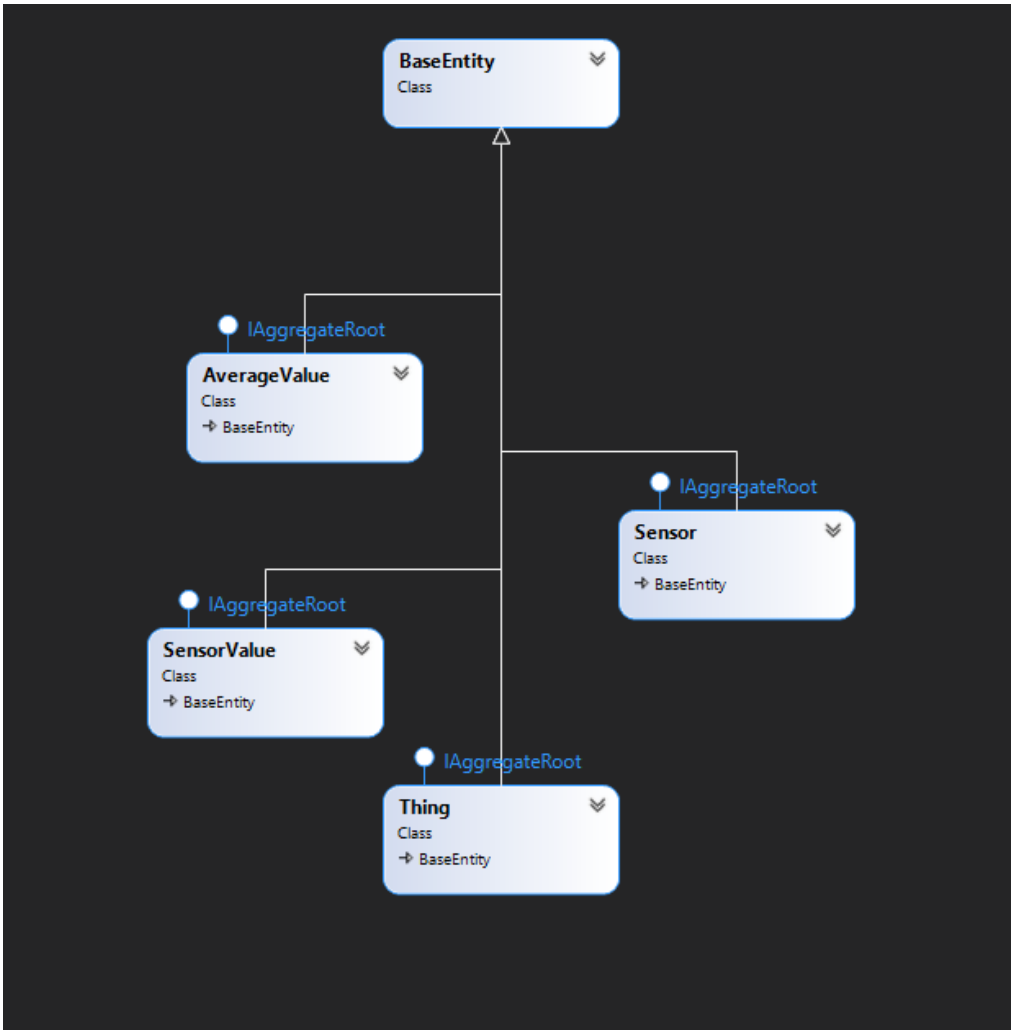

Рисунок 2.10 – Діаграма класів IoTMonitoring.Core

Проект IoTMonitoring.Data містить логіку взаємодії з базою даних:
1) EfRepository – Містить логіку взаємодії з базою даних;
2) ThingsContext – Описує які об'єкти потрібні при взаємодії бази даних та ORM;
3) ThingsContextSeed – Описує логіку яка буде виконуватися при старті взаємодії з базою даних;
4) SpecificationEvaluator – Описує логіку яка дозволяє конструювати запроси до бази даних за допомогою ORM.

5) SensorValueAggregateService – Містить логіку обробки показників сенсорів.

На рисунку 2.11 можливо побачити взаємозв'язок класів у проекті IoTMonitoring.Data.

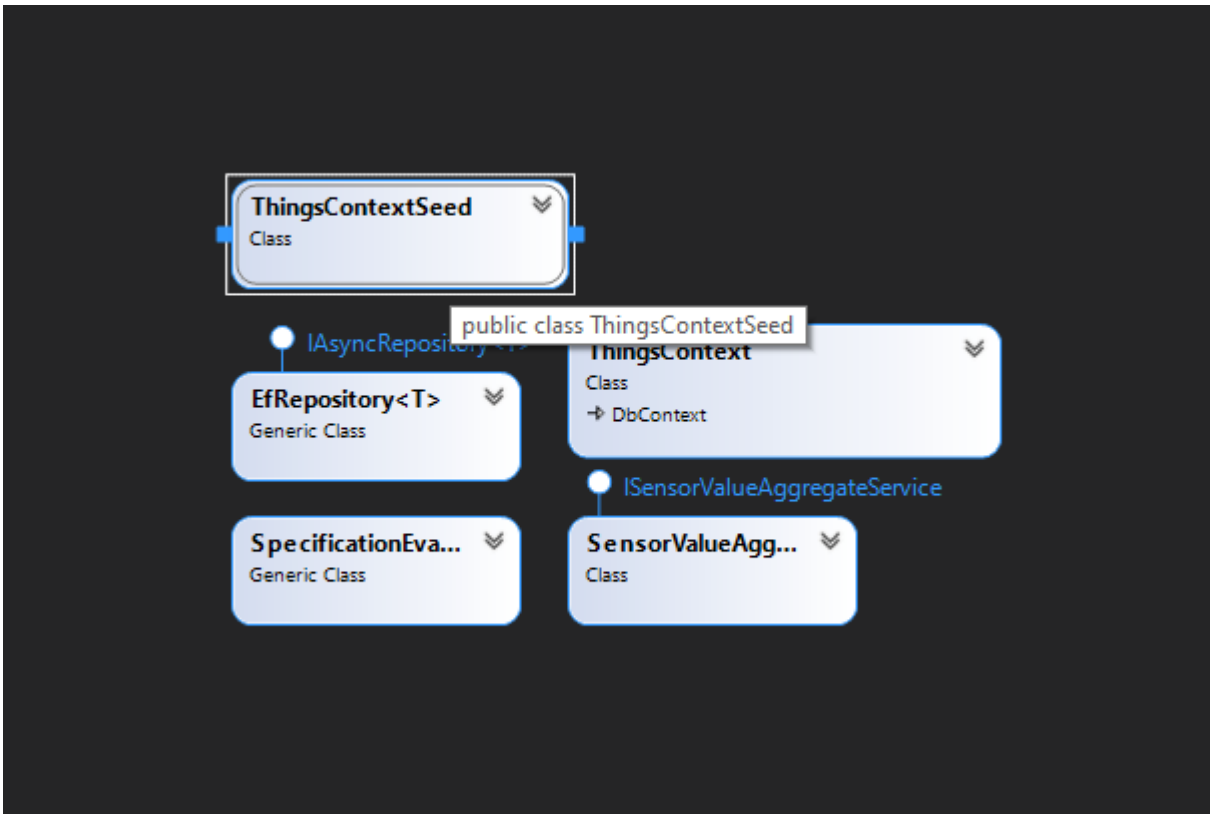

Рисунок 2.11 – Діаграма класів у IoTMonitoring.Data

Діаграма розгортання показує обчислювальні вузли під час роботи програми а також компоненти та об'єкти що виконуються можливо побачити на рис 2.12.

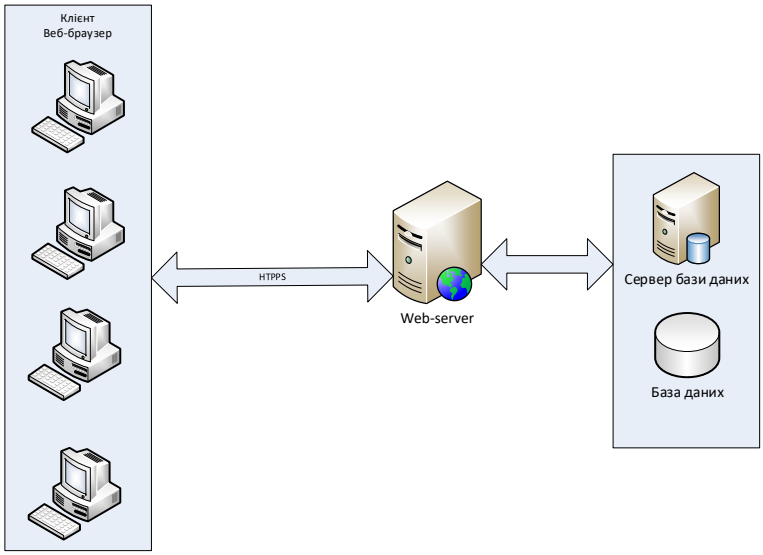

Рисунок 2.12 - Діаграма розгортання програмного забезпечення

2.6 Діаграми взаємодії

Для більш детального розуміння взаємодії безлічі об’єктів і відношень між потрібно побудувати діаграми взаємодії. Діаграми взаємодії можуть будь представлені у вигляді діаграм кооперації, послідовності і діяльності.[13]

Для деталізації взаємодії підсистем розроблюваних підсистем, користувачів та БД потрібно побудувати діаграми активності підсистем.

Діаграма активності підсистеми авторизації користувача в системі представлена на рисунку 2.13

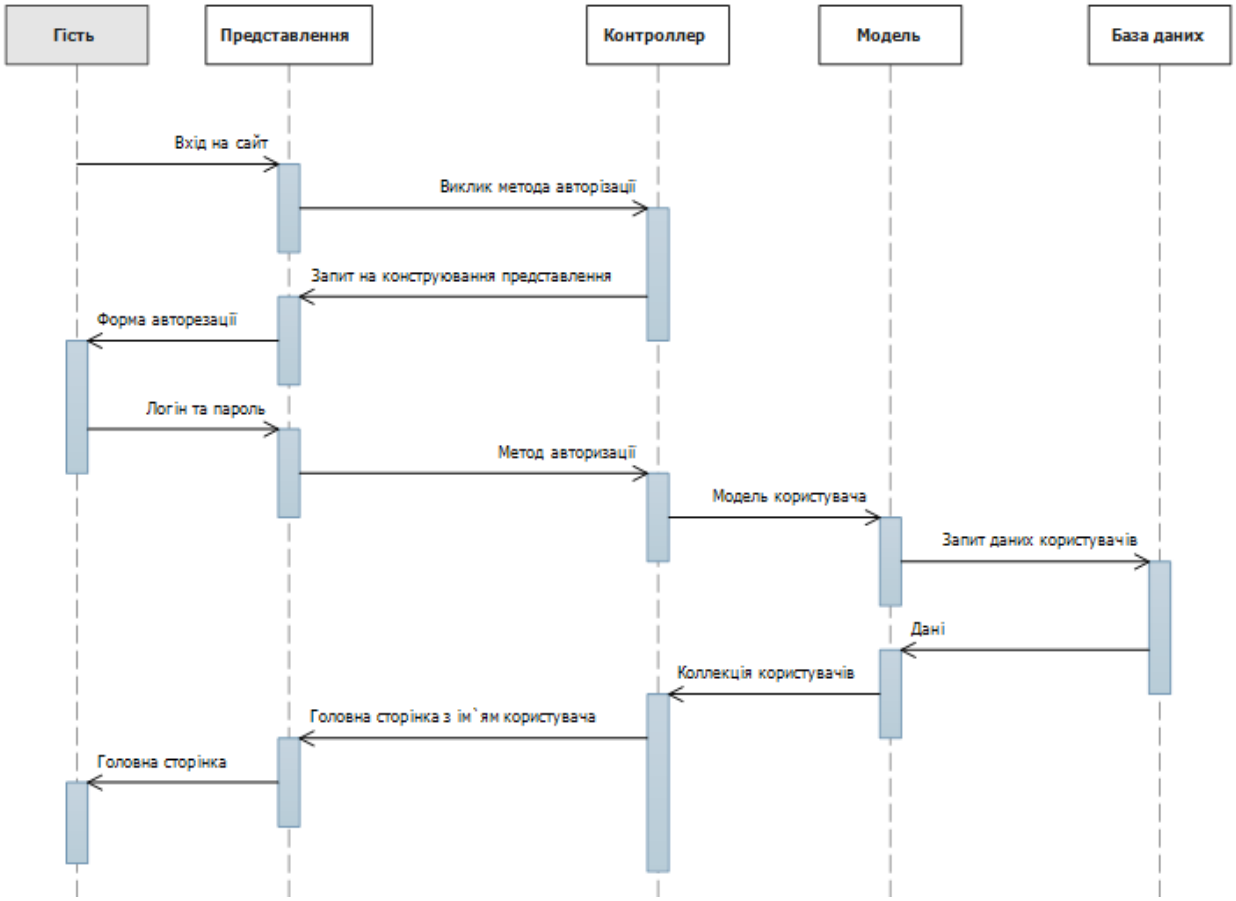

Рисунок 2.13 – Діаграма активності авторизації користувача

Діаграма активності додавання менеджером сенсору представлена на рисунку 2.14

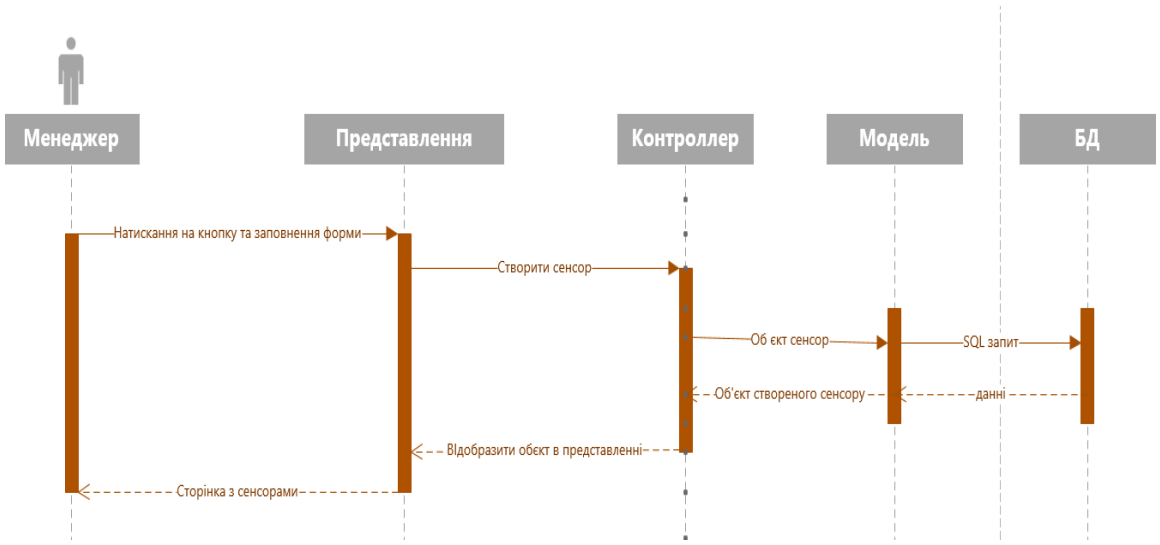

Рисунок 2.14 – Діаграма активності

Алгоритм це певна послідовність дій які виконують різні дії наприклад обробку даних, формування зображення і тп. Алгоритм має формальне виконання яке дозволяє виконувати дії та команди як людині так і різноманітним виконуючим технічним пристроям.

Система команд виконавця (СКВ) це безліч команд які в змозі виконати пристрій. Алгоритм може бути виконаний в тому випадку якщо команди входять в СКВ.

Кожен алгоритм має певні характеристики а саме:

1) зрозумілість – характеристика що визначає використання алгоритмом функції зрозумілі виконавцю;
2) дискретність – характеристика що визначає що алгоритм повинен надавати рішення як послідовне виконання простих дій;
3) детермінованість – характеристика що визначає те що кожний окремий крок алгоритму має однозначно визначатися станом системи;
4) універсальність – характеристика що визначає те що алгоритм повинен к різноманітним вхідним даним;
5) результативність – характеристика що визначає те що алгоритм повинен завершуватися певними результатами.

Виконавцем алгоритму є деяка реальна або абстрактна (технічна або біологічна) система що здатна виконати дії, передбачені алгоритмом.

Виконавця характеризує:

1) середа;
2) система команд;
3) відмови;
4) елементарні дії.

Система команд визначає певний кінцевий список команд який виконавець має змогу виконувати. У кожної команди є умови застосовності та описання

результату виконання програми. Після того як була викликана команда виконавець робить відповідну елементарну дію. Середа це місце виконання алгоритму виконавцем. Відмови виникають якщо команда викликається при неприпустимому стані середи.

Алгоритми можуть бути задані декількома засобами, а саме:

1) словесний опис алгоритму;
2) графічний опис алгоритму;
3) задання за допомогою псевдокоду;
4) задання за допомоги мови програмування.

Графічне завдання алгоритму виконується за допомогою блок-схем. Блок схема надає представлення алгоритму за допомогою геометричних фігур, які називаються блоками. Послідовність блоків і ліній що їх сполучають утворюють блок-схему. Команди алгоритму у вигляді формального опису або коду поміщають всередину блоків, блоки з'єднують стрілками які показують послідовність виконання команд алгоритму. Блок схеми мають значно більшу наочність ніж словесний опис алгоритму, але ця наочність втрачається при зображенні великих алгоритмів. Альтернативним методом графічного опису є створення ER-схем.

Опис за допомогою псевдокоду є більш схожим програмування на язику високого рівня. Цей метод є менш виразним ніж графічний, за допомогою його складно проектувати зв'язки з управлінням, і він є нестандартизованим.

Нижче представлено опис алгоритмів за допомогою блок схем.

Для реалізації функції додавання нового сенсору в систему необхідно, перейти на сторінку сенсорів , визвати форму додавання сенсору, заповнити поля форми, виконати запит до бази даних, отримати результати запиту та вивести отриманий сенсор в загальному відображенні сенсору. Блок-схема алгоритму додавання сенсору представлена на рисунку 20

Для реалізації функціоналу сбіру показників з сенсору потрібно в програмному забезпеченні по сбіру інформації з обладнання додати вузол сбіру даних за необхідним протоколом(напр. Modbus) та приєднати його до вузлу відправки інформації до ПЗ по обробці, при цьому при налаштуванні вузла потрібно вказати сенсор із доступних, після цього почнеться сбір показників цього сенсору. Блок-схема алгоритму збирання даних представлена на рисунку 2.15 та рисунку 2.16

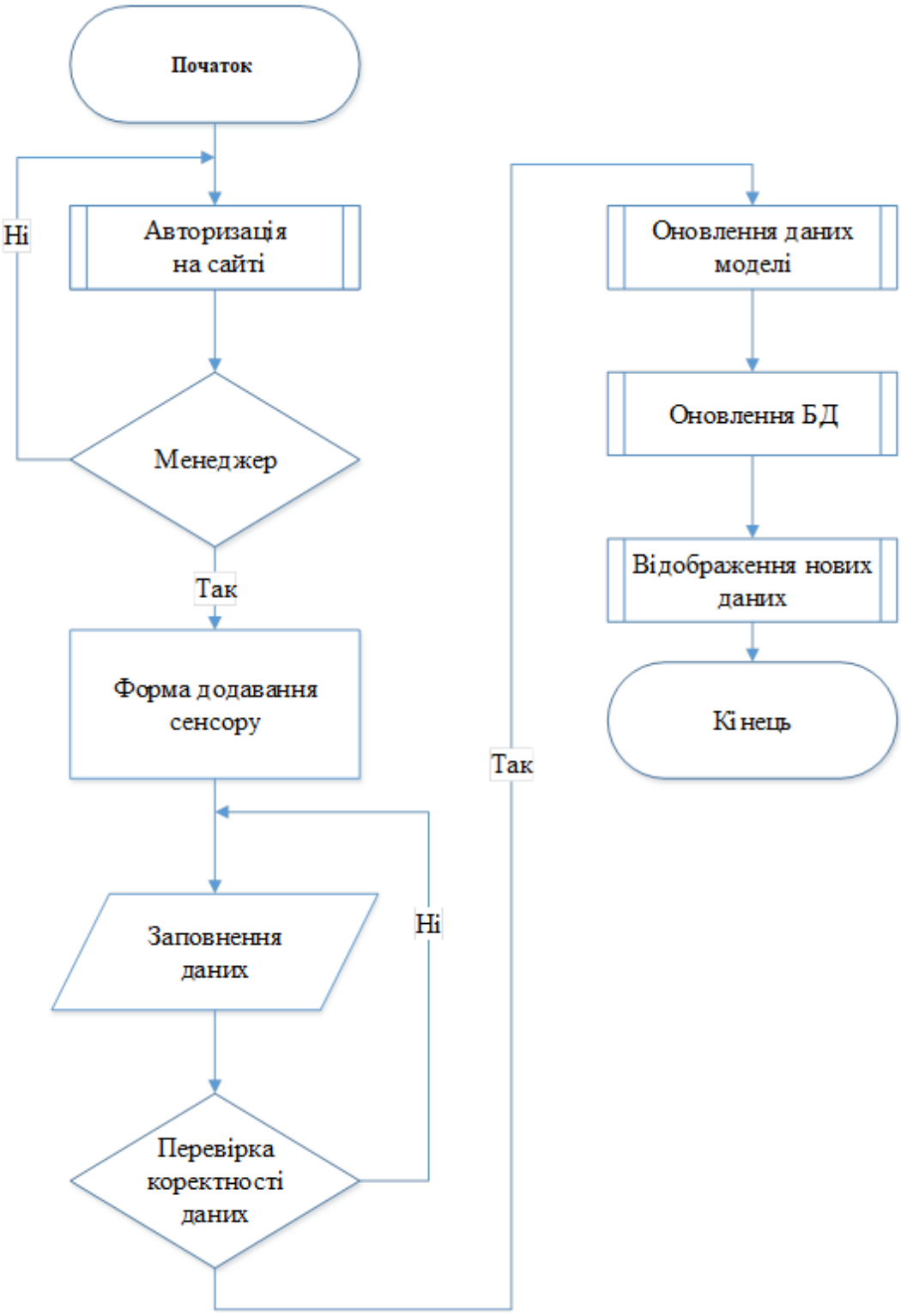

Рисунок 2.15 – Блок-схема алгоритму додавання сенсору

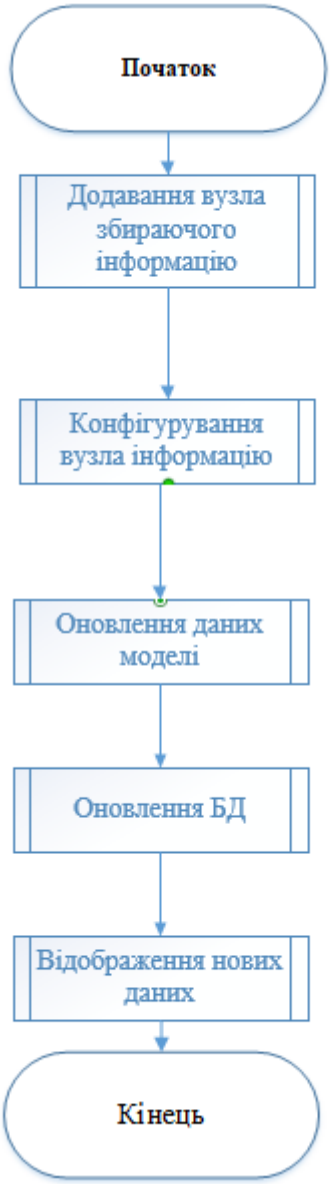

Рисунок 2.16 – Блок-схема алгоритму збирання даних

Клієнтська частина виконана за допомогою фреймворку Angular 8. Клієнтська частина представляє собою Single Page Application(SPA). Для отримання повідомлень з серверу додатків використовується SignalR. Схема взаємодії с сервером за допомогою SignalR показана на рисунку 2.17.

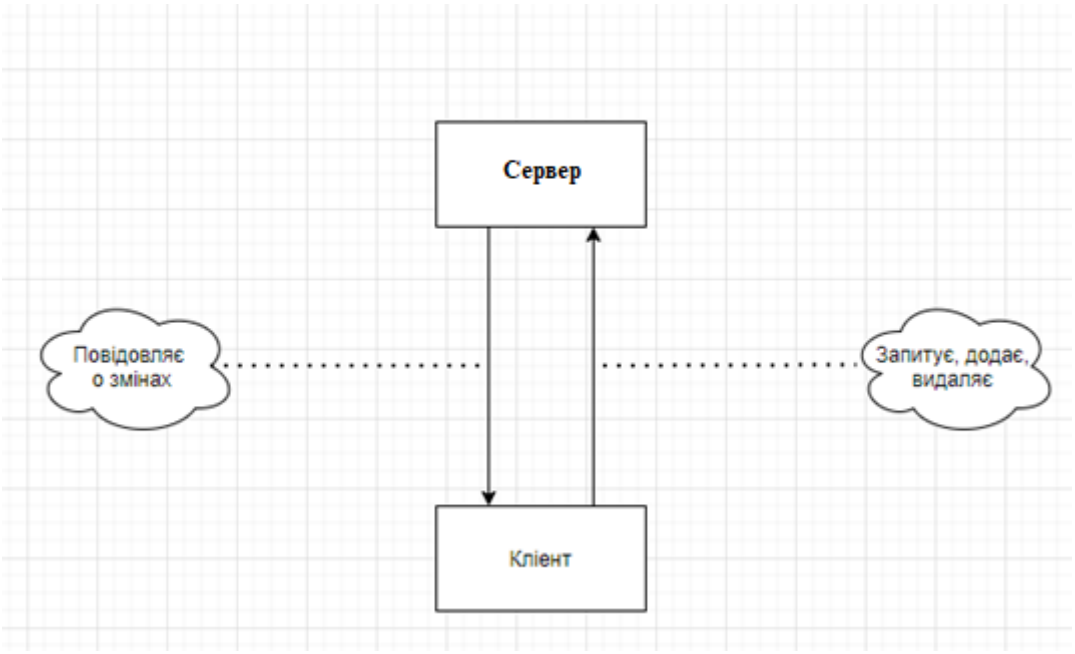

Рисунок 2.17 – Схема взаємодії с сервером за допомогою SignalR

За допомогою SignalR реалізовано оновлення графіків та спектрограми. Клієнтська частина надає користувачам можливість переглядати як історичні дані про вібрації так і поточні показники. Приклад спектрограми яка відображає як історичні показники так і показники у реальному часі можливо побачити на рисунку 2.18.

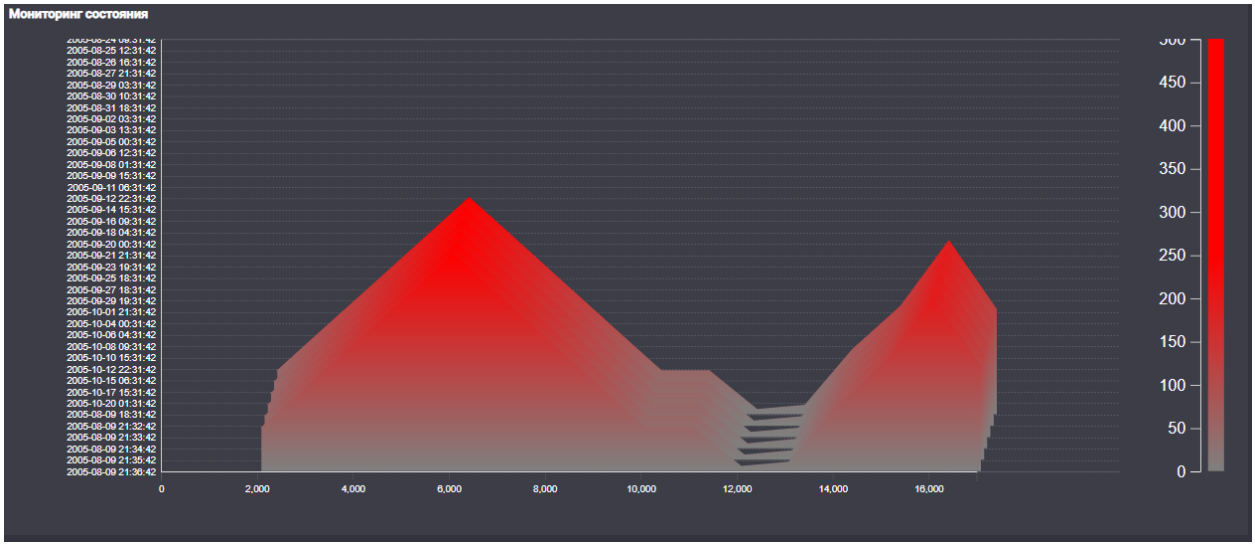

Рисунок 2.18 – Спектрограма з історичними даними

2.7 Алгоритм оцінки стану обладнання

При розробці алгоритму оцінки стану в першу чергу потрібно було досягти мінімальну потрібність алгоритму у навчанні. Тобто щоб оцінка стану проходила точно при мінімальній потрібності у розмічених даних. Для цього алгоритм потрібен мати змогу ідентифікувати патерни та до навчатись на вхідних нерозмічених даних. Подібні техніки називають System Identification.

Приймаючі до уваги вищеописані вимоги та аналіз поточного стану проблеми визначеного у першому розділі за основу був прийнятий алгоритм на базі Dictionary Learning.

Для реалізації алгоритму була використана мова програмування Python так як вона має значну бібліотеку засобів для аналізу сигналів, роботи з нейронними мережами та багату функціональність. Так як при побудуванні системи ми дотримувалися мікро сервісної архітектури то алгоритм оцінки стану обладнання також представлений у вигляді окремого сервісу. Це дозволяє легко та зручно масштабуватись, додаючи нові екземпляри сервісу. Сервіс оцінки не зберігає стану залежного від інших сервісів тому таких екземплярів може бути скільки завгодно екземплярів при тому не потрібно синхронізувати стан цих екземплярів. Кожен екземпляр сервісу знаходиться у окремому контейнері Docker. Це дозволяє швидко розгортувати екземпляри сервісу як на Windows, Linux так і в кластерах з Kubernetes.

При розробці алгоритму використовувався JupyterLab. Це інтерактивне веб-середовище розробки для блокнотів, коду і даних Jupyter. JupyterLab відрізняється гнучкістю: налаштовуйте і упорядковуйте призначений для користувача інтерфейс для підтримки широкого спектра робочих процесів в області науки про дані, наукових обчислень і машинного навчання. JupyterLab є розширюваним і модульним. Є можливість писати плагіни, які додають нові компоненти і інтегруються з існуючими. Побачити інтерфейс JupyterLab можливо на рисунку 2.19.

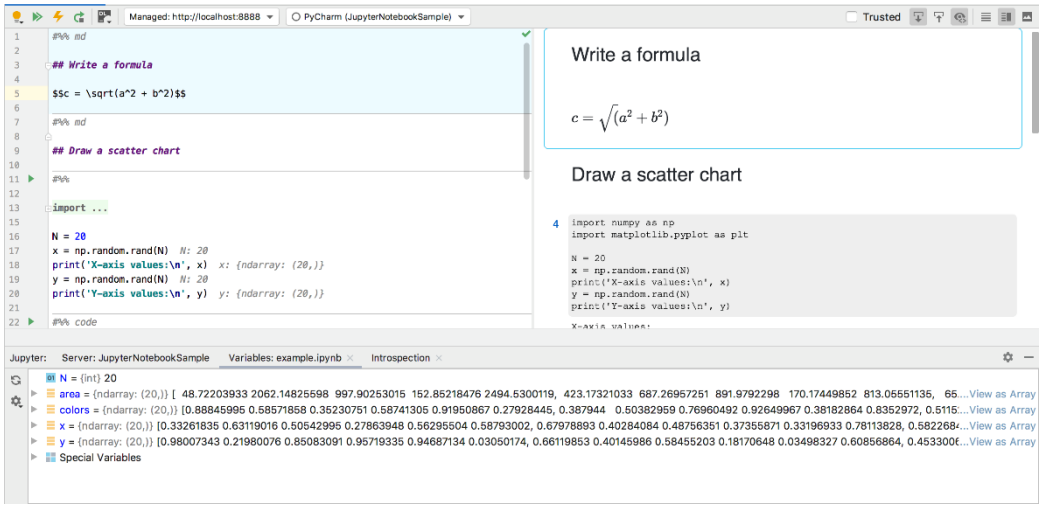

Рисунок 2.19 – Інтерфейс JupyterLab

Перший рівень ідентифікації при проведенні вібраційної діагностики це виявлення наявності дефекту. Для виявлення наявності дефекту був використаний алгоритм заснований на аналізу форми тональності. Цей метод найшвидший та потребує найменш ресурсів. Якщо цей метод виявив дефект починають виконуватися наступні алгоритми для класифікації та локалізації пошкоджень.

Блок схема алгоритму виявлення наявності дефектів представлена на рисунку 2.20.

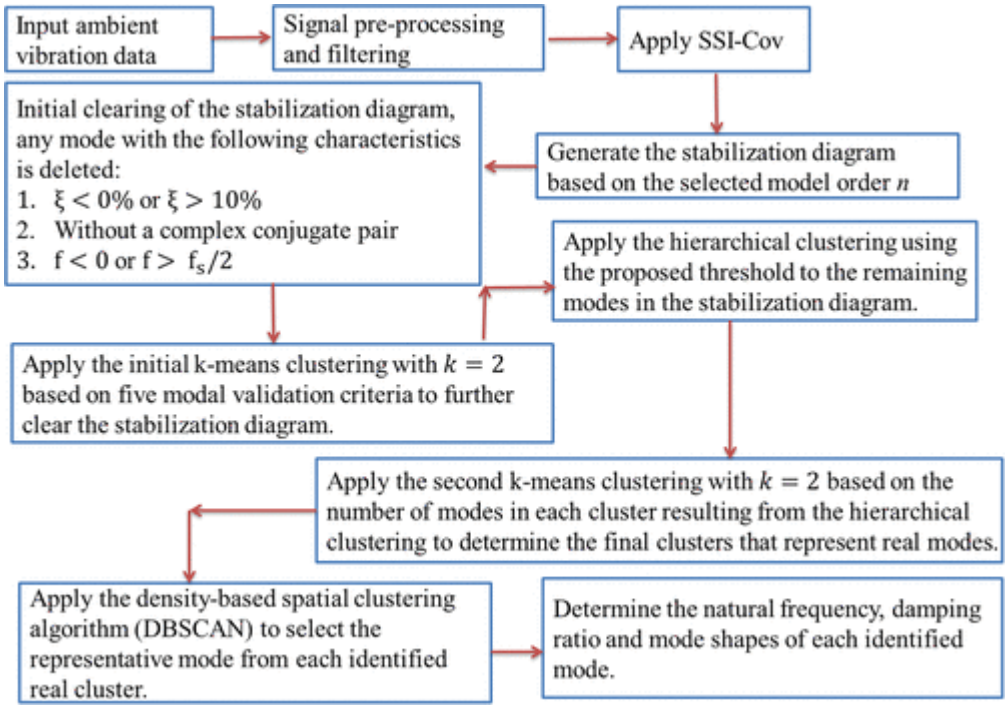

Рисунок 2.20 – Блок схема алгоритму

Якщо алгоритм виявив наявність дефекту то наступним кротом буде його класифікація. Для класифікації пошкоджень був використаний алгоритм на базі Dictionary Learning. Блок схема алгоритму для класифікації пошкоджень представлена на рисунку 2.21.

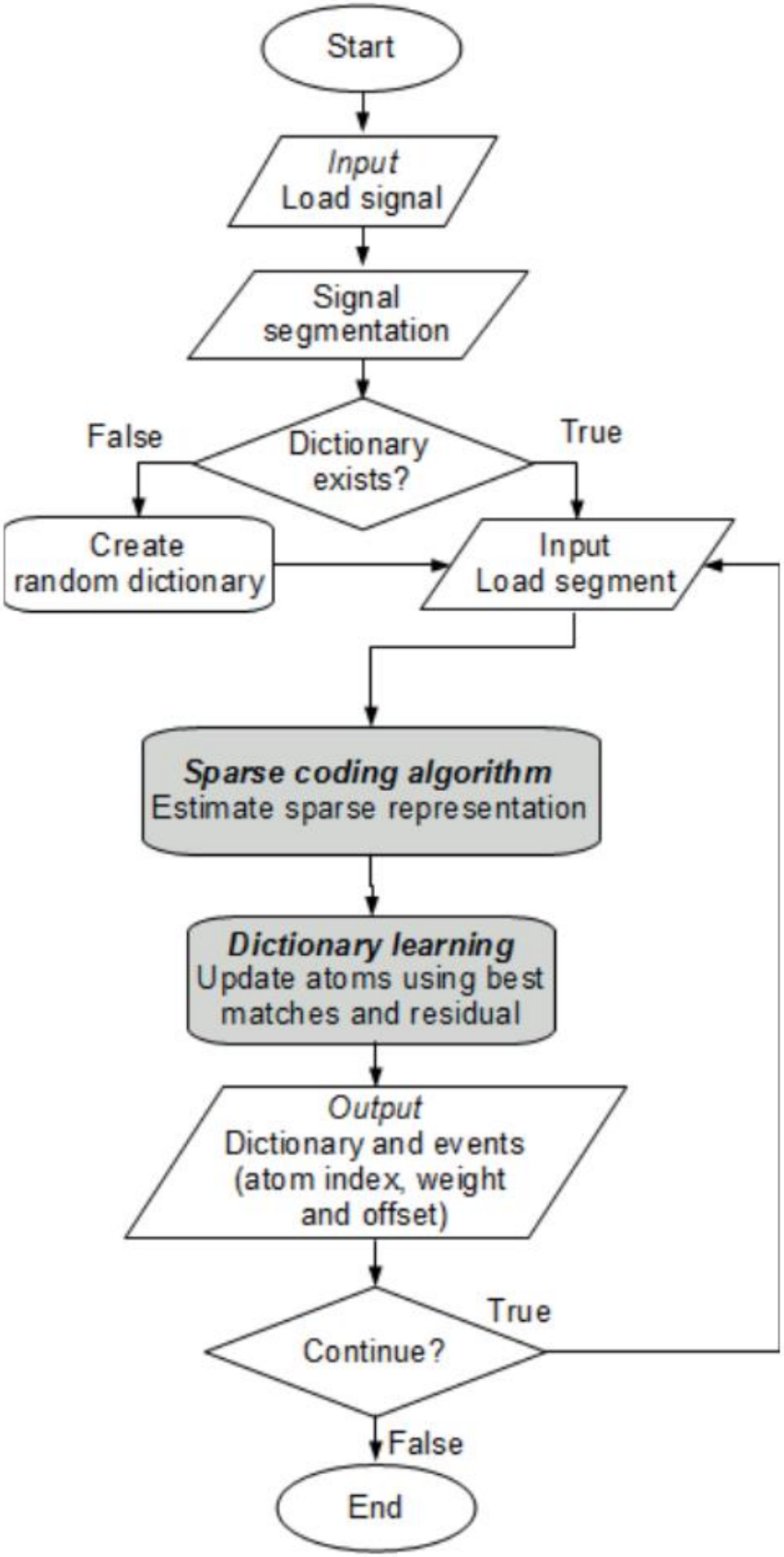

Рисунок 2.21 – Блок схема алгоритму класифікації пошкоджень

2.8 Збір показників з сенсорів вібраційного прискорення

Є кілька варіантів доставки показників швидкості в інформаційну систему. Найбільш доцільним являється повторне використання коду доставки повідомлень датчика вібрації. В якості апаратної платформи використовується CC2650 Launchpad, який зображено на рисунку 2.22:

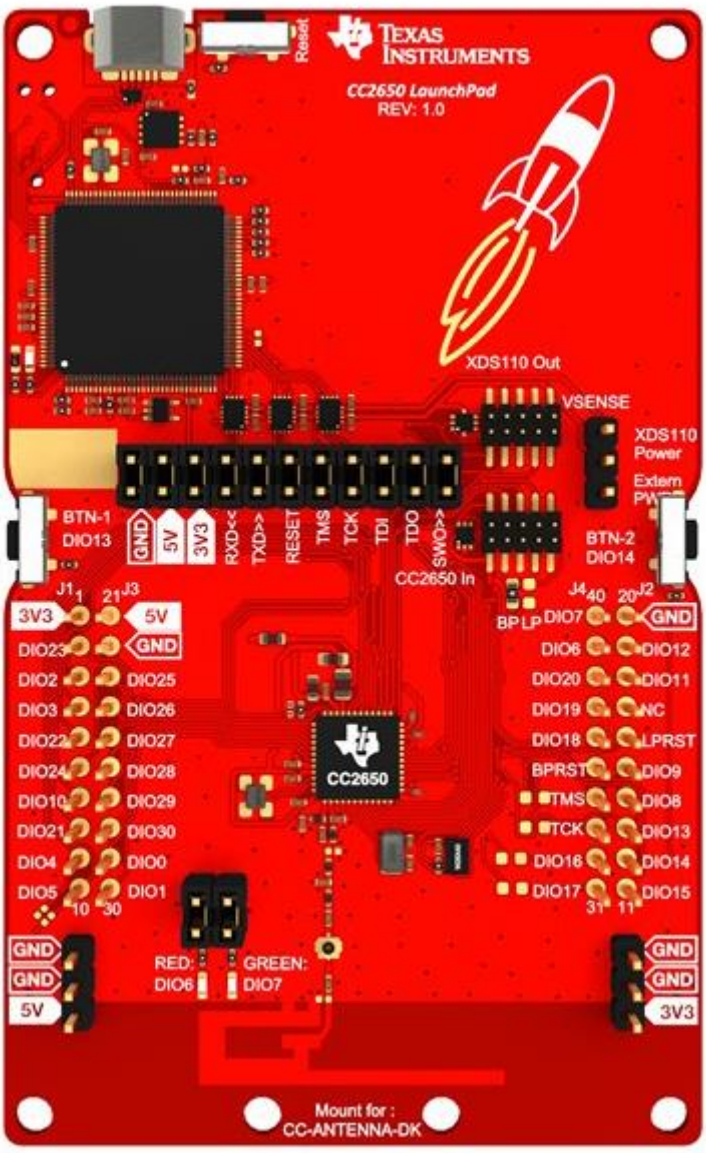

Рисунок 2.22 – CC2650 Launchpad

Розробка програмно-апаратних засобів стенду включає проектування шаблону взаємодії суті уявлення і суті постачальника даних. Відповідним шаблоном буде Модель–вигляд–контролер.

Цей шаблон передбачає поділ системи на три взаємопов'язані частини: модель даних, вигляд (інтерфейс користувача) та модуль керування. Застосовується для відокремлення даних (моделі) від інтерфейсу користувача (вигляду) так, щоб зміни інтерфейсу користувача мінімально впливали на роботу з даними, а зміни в моделі даних могли здійснюватися без змін інтерфейсу користувача.

Мета шаблону — дизайн ПЗ, який повинен полегшувати подальші зміни чи розширення програм, а також надавати можливість повторного використання окремих компонентів програми. Крім того використання цього шаблону у великих системах сприяє впорядкованості їхньої структури і робить їх більш зрозумілими за рахунок зменшення складності.

У рамках архітектурного шаблону модель–вигляд–контролер (MVC) програма поділяється на три окремі, але взаємопов'язані частини з розподілом функцій між компонентами.

Структура отриманого проекту представлена на рисунку 2.23:

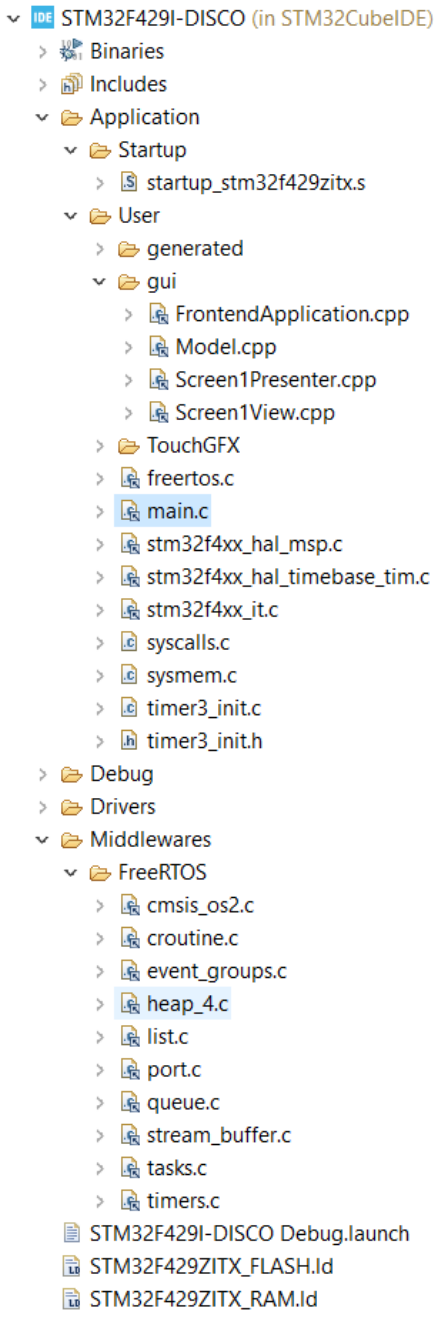

Рисунок 2.23 – Склад проекту стенду

Взаємодія з інформаційною системою відбувається за допомогою протоколу BLE. Архітектура мережевого взаємодії побудована на основі реалізації основного контролера в ролі хоста і BLE системи у вигляді мережевого контролера. Взаємодія реалізовано з використанням протоколу USART. Реалізація взаємодії представлена на рисунку 2.24:

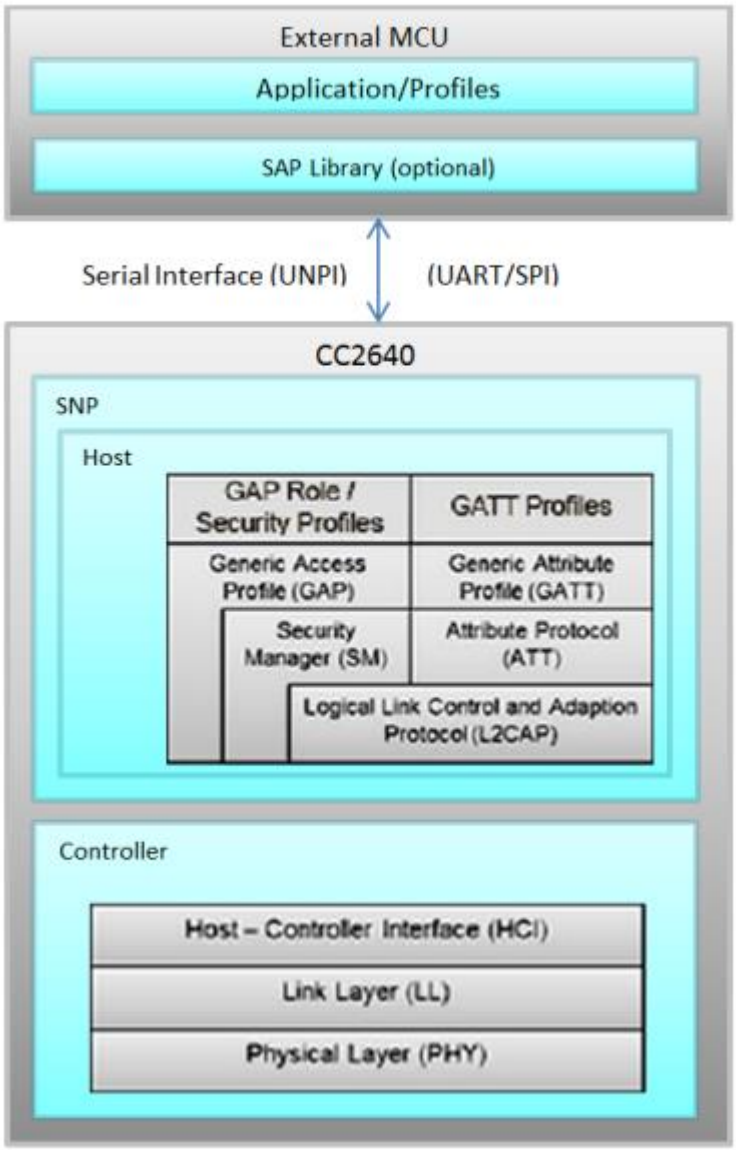

Рисунок 2.24 – Реалізація взаємодії основного контролеру та BLE контролеру

З боку мережевого контролера необхідно реалізувати розпакування пакетів і взаємодія з HUB рівнем. Склад системи розробки під мережевий контролер представлений на рисунку 2.25:

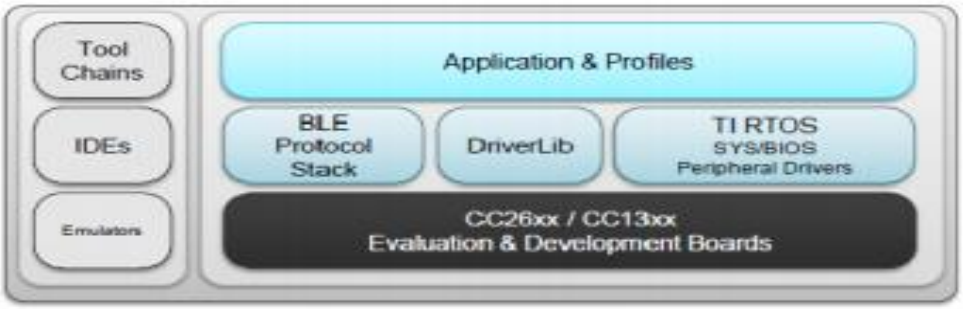

Рисунок 2.25 – Склад системи розробки під CC2650

Основні компоненти системи розробки:

1) Операційна система в режимі реального часу (RTOS) з ядром TI-RTOS SYS / BIOS;
2) CC26xxware driverLib забезпечує рівень абстракції регістрів і використовується програмним забезпеченням та драйверами для CC2650 SoC;
3) Стек протоколів низької енергії Bluetooth надається у вигляді бібліотеки з частинами стеку протоколів в ПЗУ CC2640.

Зразки додатків та профілів полегшують початок розробки, використовуючи як власні, так і загальні рішення. Основний шаблон проекту містить вихідний код як для стеку протоколу, так і для прикладного рівню. Склад шаблону проекту зображено на рисунку 2.26:

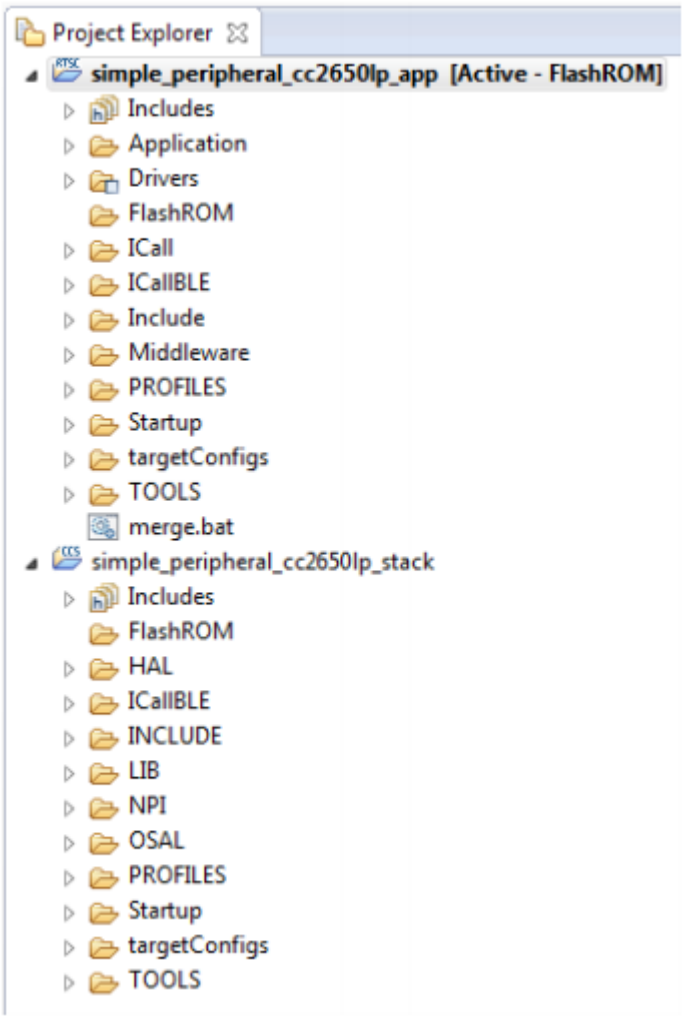

Рисунок 2.26 – Склад шаблону проекту

Проект стека включає нижні шари стека протоколу BLE а також включаючи рівні GAP та GATT. Більшість інших виробників надає код стеку як бібліотеку. Проект прикладного рівня включає профілі, код програми, драйвери та модуль ICall.

ICall_init () ініціалізує екземпляр модуля ICall, а ICall_createRemoteTasks () створює завдання на зовнішнє зображення з функцією введення за відомою адресою. Після ініціалізації ICall, завдання програми реєструється в ICall через ICall_registerApp. Після запуску планувальника SYS / BIOS і додаток Завдання виконується, програма надсилає команду протоколу, визначену в ICallBLEAPI.c, таку як GAP_GetParamValue(). Команда протоколу не виконується в потоці програми, а інкапсульована в повідомлення ICall і направлено до завдання стеку протоколу через фреймворк ICall. Ця команда надсилається диспетчеру ICall, де вона відправляється та виконується на стороні сервера (тобто стек низької енергії Bluetooth). Потік програми тим часом блокується (тобто чекає) для відповідного повідомлення про стан команди (тобто статус і значення параметра GAP). Коли стек протоколів низького енергоспоживання Bluetooth завершує виконання команди, повідомлення про стан команди відповідь надсилається через ICall назад у потік програми. Приклад взаємодії представлений на рисунку 2.27:

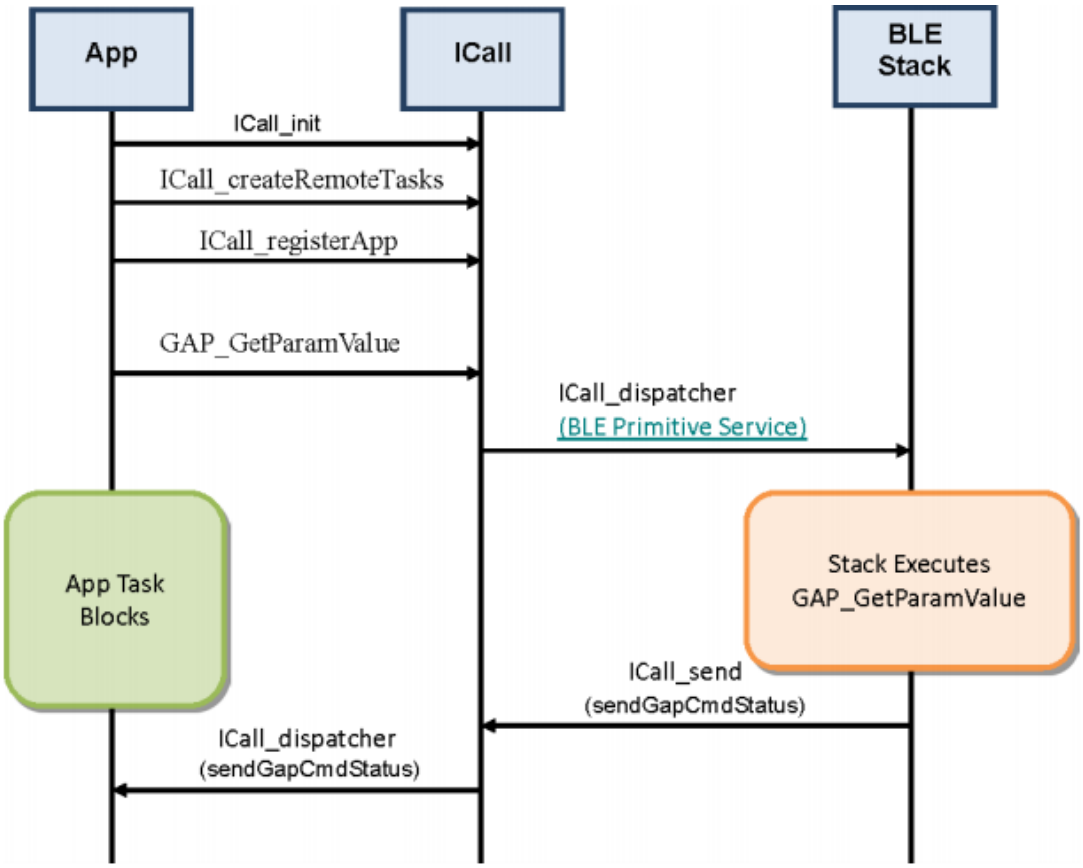

Рисунок 2.27 – Використання інтерфейсу ICall

Сам додаток обробляє пакети від основного контролера при в окремому потоці. Хід виконання потоку представлений на рисунку 2.28:

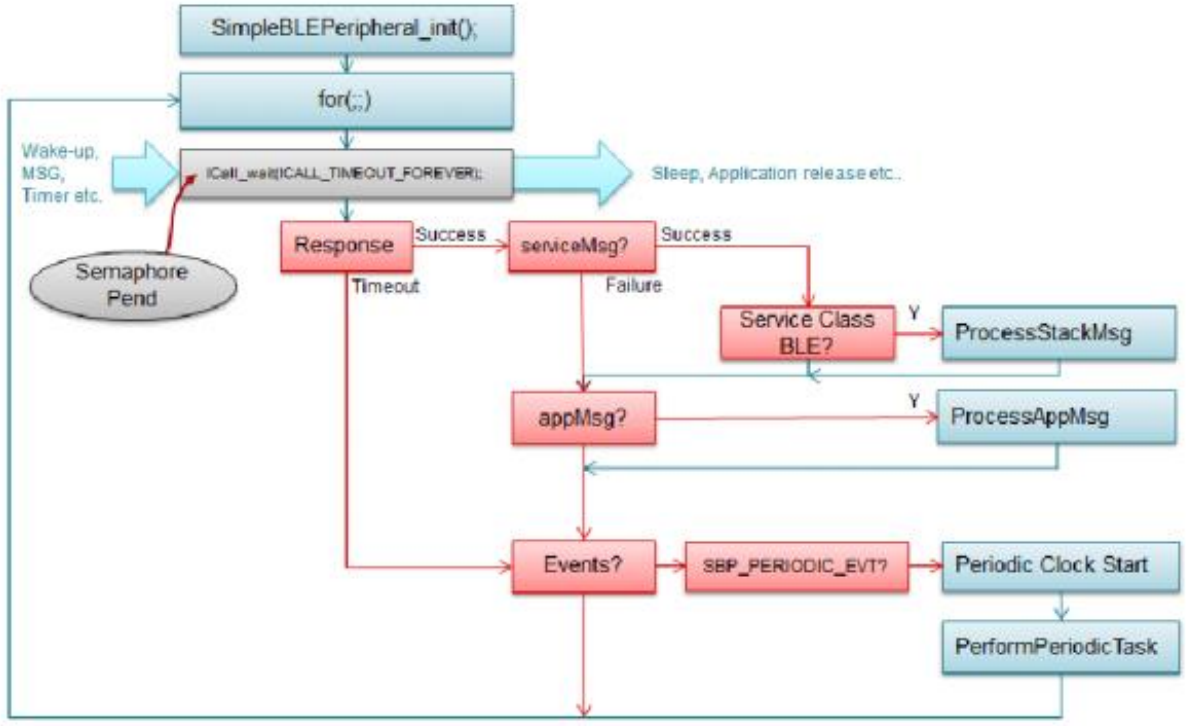

Рисунок 2.28 – Алгоритм обробки потоків мережевим контролером

Постачальник STK розробив рівень GATT стека протоколу Bluetooth Low Energy для використання додатком для даних зв'язок між двома підключеними пристроями. Дані передаються та зберігаються у формі характеристики, які зберігаються в пам'яті на пристрої Bluetooth Low Energy. У GATT коли два пристрої підключені, кожен виконує одну з двох ролей:

1) сервер GATT - Цей пристрій містить характерну базу даних, яку читає або записує GATT клієнт;
2) клієнт GATT - цей пристрій зчитує або записує дані із сервера GATT або на нього.

На це показано взаємозв'язок у зразку з'єднання Bluetooth Low Energy, де периферійний пристрій. Реалізація взаємодії продемонстровано на рисунку 2.29:

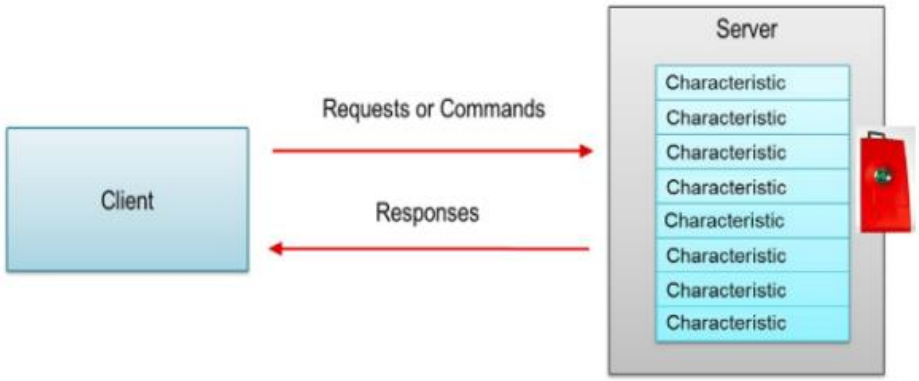

Рисунок 2.29 – Взаємодія з стороннім пристроєм

Хоча характеристики іноді взаємозамінні, коли йдеться про Bluetooth Low Energy, враховуйте їх як групи інформації, що називаються атрибутами. Атрибути - це основні групи переданої інформації між пристроями. Характеристики організовують і використовують атрибути як значення даних, властивості та конфігурацію. Типова характеристика складається з таких атрибутів:

1) значення характеристики: Це значення - це значення даних характеристики;
2) декларація характеристик: дескриптор зберігає властивості, розташування та тип характеристики значення;
3) конфігурація характеристики клієнта: Ця конфігурація дозволяє серверу GATT налаштувати характеристику для пересилання на сервер GATT (повідомлення) або надсилання на сервер GATT і очікування підтвердження;
4) опис користувача характеристики: Цей опис являє собою рядок ASCII, що описує характеристику. Ці атрибути зберігаються на сервері GATT у таблиці атрибутів;
5) дескриптор - Ця властивість є індексом атрибута в таблиці. Кожен атрибут має унікальний дескриптор;
6) тип - Цей атрибут вказує, що представляють дані атрибута. Цей атрибут називається універсальним унікальний ідентифікатор (UUID). Деякі з цих UUID визначаються Bluetooth SIG, а інші визначаються користувачем.

2.9 Діаграми які описують систему вібраційної діагностики

Загальна діаграма взаємодії представлена на рисунку 2.30.

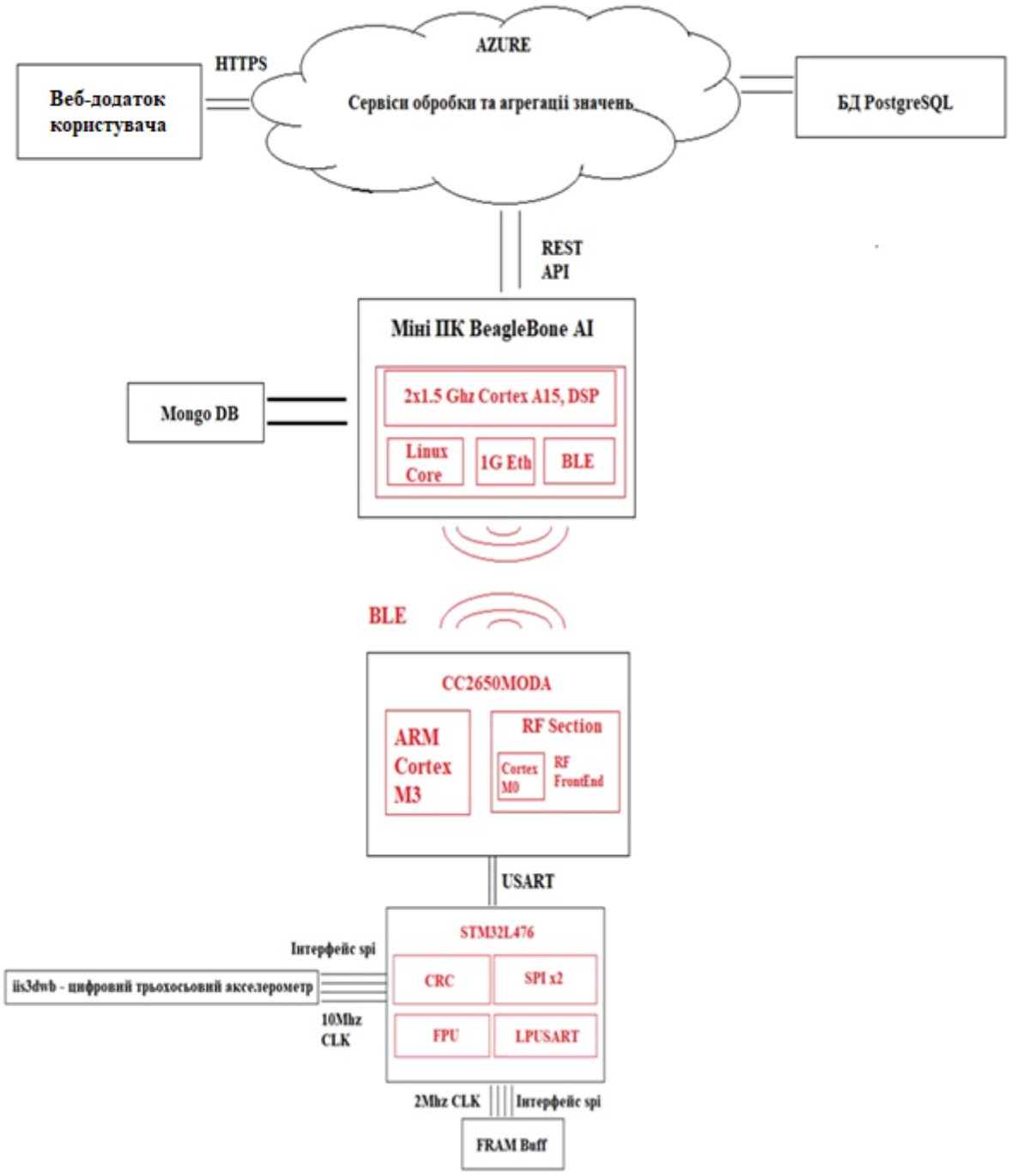

Рисунок 2.30 – Загальна діаграма взаємодії

Діаграма яка описує функціональність частин системи вібраційної діагностики показана на рисунку 2.31.

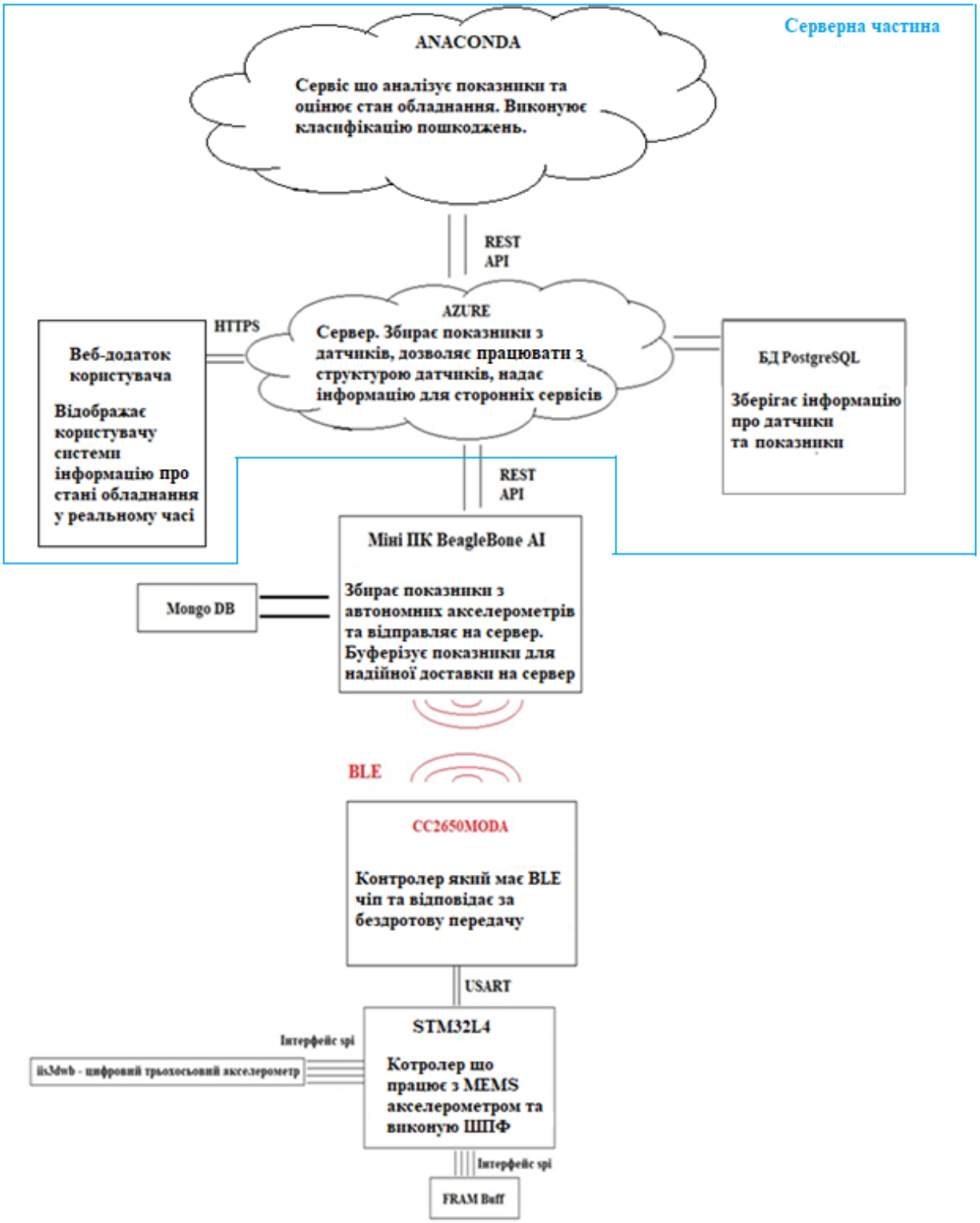

Рисунок 2.31 – Діаграма описуюча функціональність системи вібраційної діагностики

2.10 Висновки за другим розділом

У результаті роботи над другим розділом був запланований експеримент по проведенню оцінки системи вібраційної діагностики, розроблена система вібраційної діагностики та розроблено алгоритм оброблення результатів експерименту. Описано інструментарій, та зіставні частини системи зокрема серверну частину. У пункті 2.4 описується розроблена БД для зберігання показників з сенсорів вібраційної діагностики. У пункті 2.7 наведено алгоритм що використовується для оцінки стану обладнання. У пункті 2.5 описана розроблена серверна частина системи вібраційної діагностики.

3 АНАЛІЗ РЕЗУЛЬТАТІВ ЕКСПЕРИМЕНТУ ПРОВЕДЕННЯ ВІБРАЦІЙНОЇ ДІАГНОСТИКИ І ФОРМУЛЮВАННЯ ПРАКТИЧНИХ РЕКОМЕНДАЦІЙ

Для оцінки придатності роботи системи був зроблений стенд що імітує електричний двигун та генератор з’єднаний муфтою. Для виготовлення стенду використовувався 3D принтер Anycubic Mega S. 3D модель корпусу представлена на рисунку 3.1.

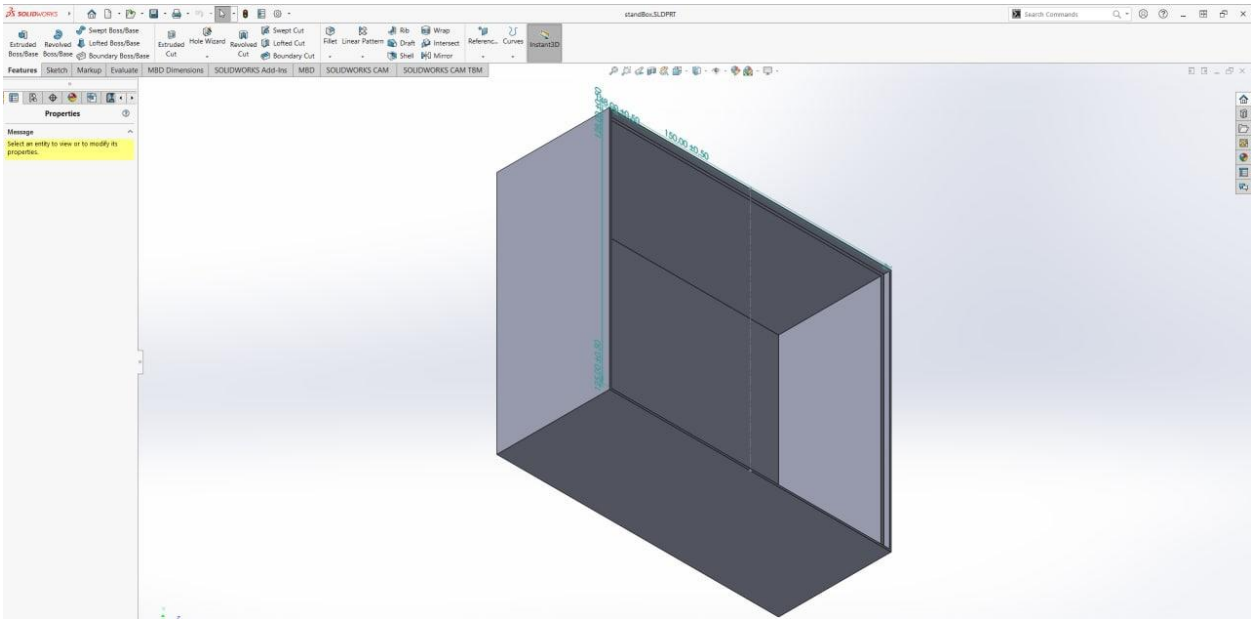

Рисунок 3.1 - 3D модель корпусу стенду

В корпус стенду закріплений блок живлення та плати керуючі двигунами один з яких імітує електродвигун, а другий генератор. На рисунку 3.2 представлені змонтовані компоненти у корпусі.

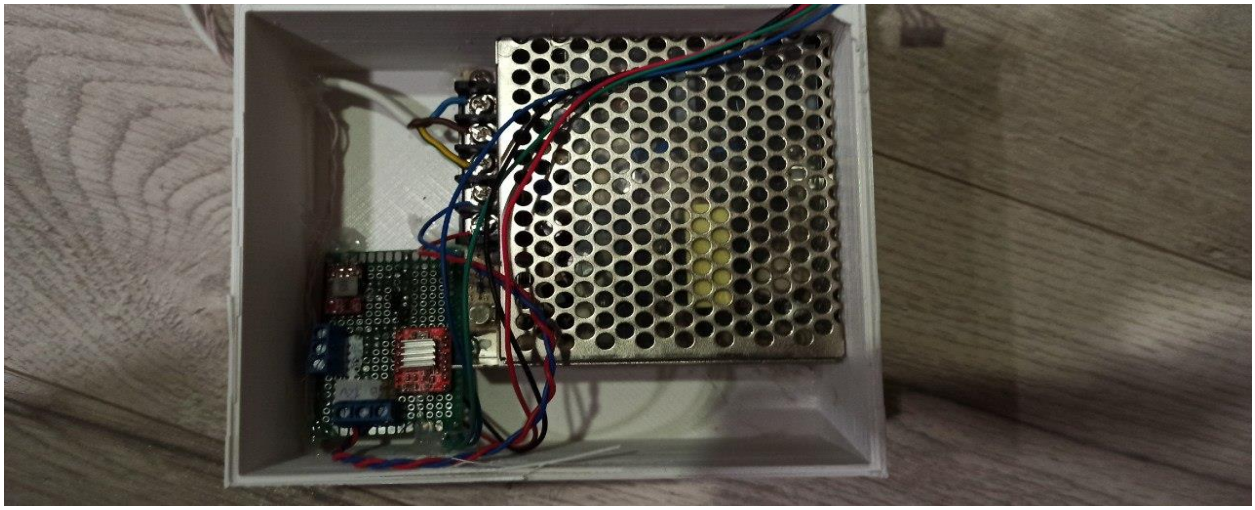

Рисунок 3.2 – Компоненти змонтовані у корпусі

Зверху на корпусі розташовані електродвигуни закріплені у корпусах. Корпус електродвигуна що імітує генератор та створює навантаження представлений на рисунку 3.3.

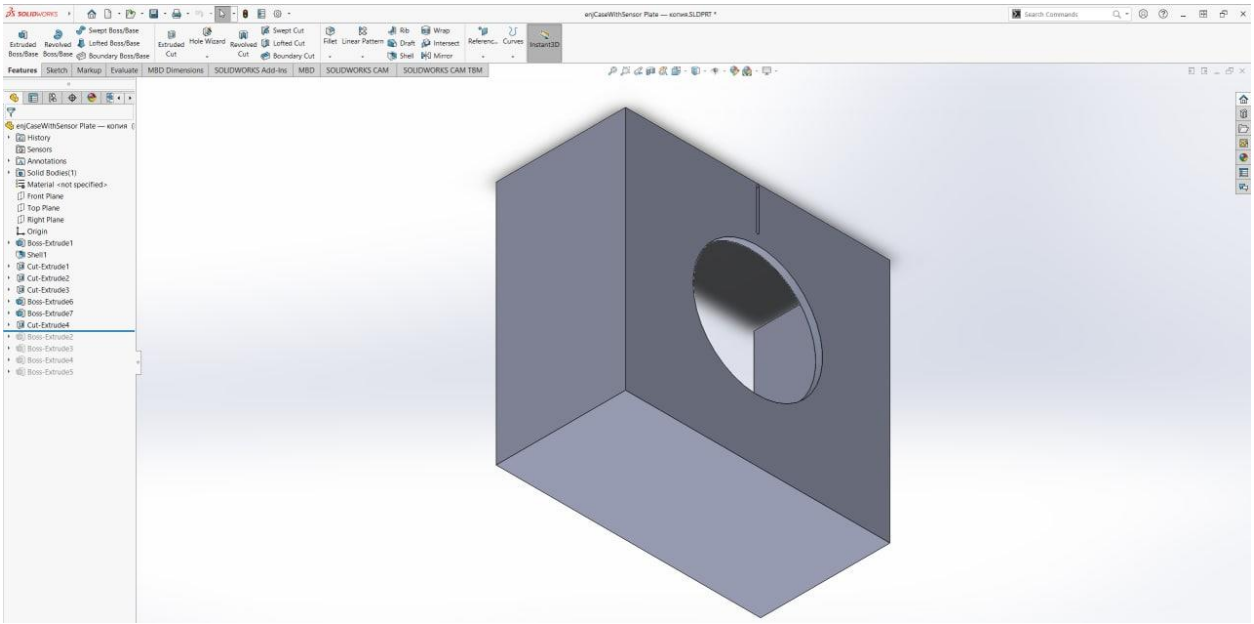

Рисунок 3.3 – Корпус електродвигуна який створює навантаження

На корпус другого двигуна який обертається закріплений вібраційний датчик. 3D модель корпусу з місцем кріплення вібраційного датчику представлена на рисунку 3.4.

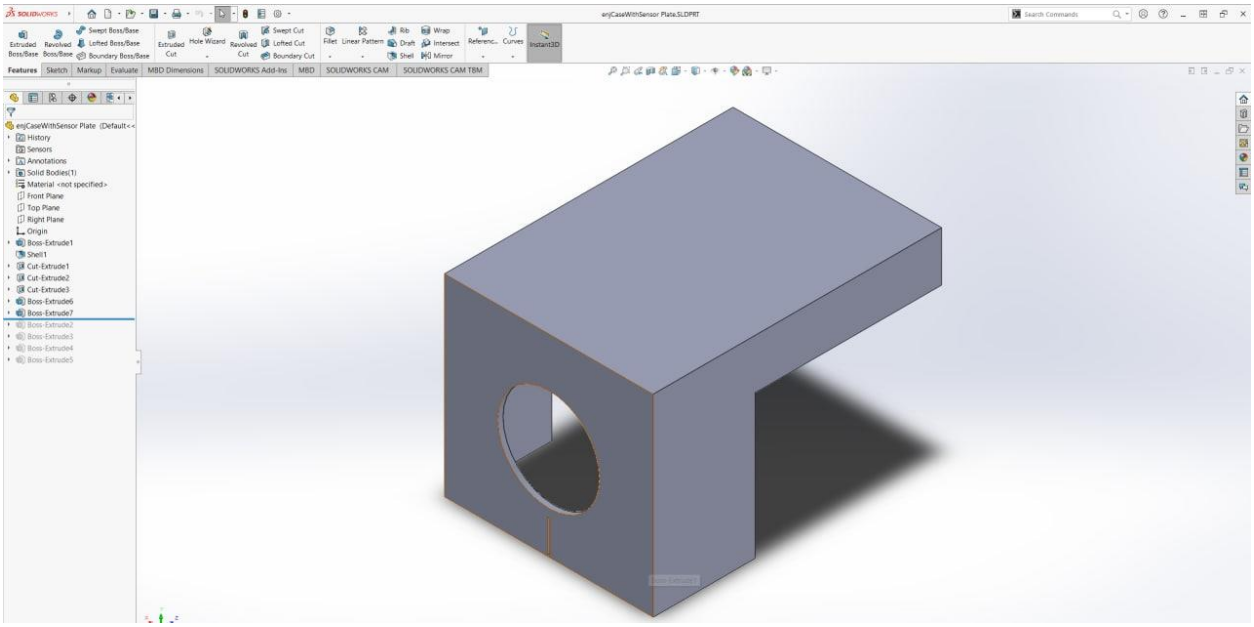

Рисунок 3.4 – Корпус електродвигуна з місцем під кріплення вібраційного датчика

Також для зручного управління стендом створена керуючій пульт з графічним інтерфейсом який дозволяє керувати обертами двигуна. 3D модель корпусу керуючого пульта представлений на рисунку 3.5.

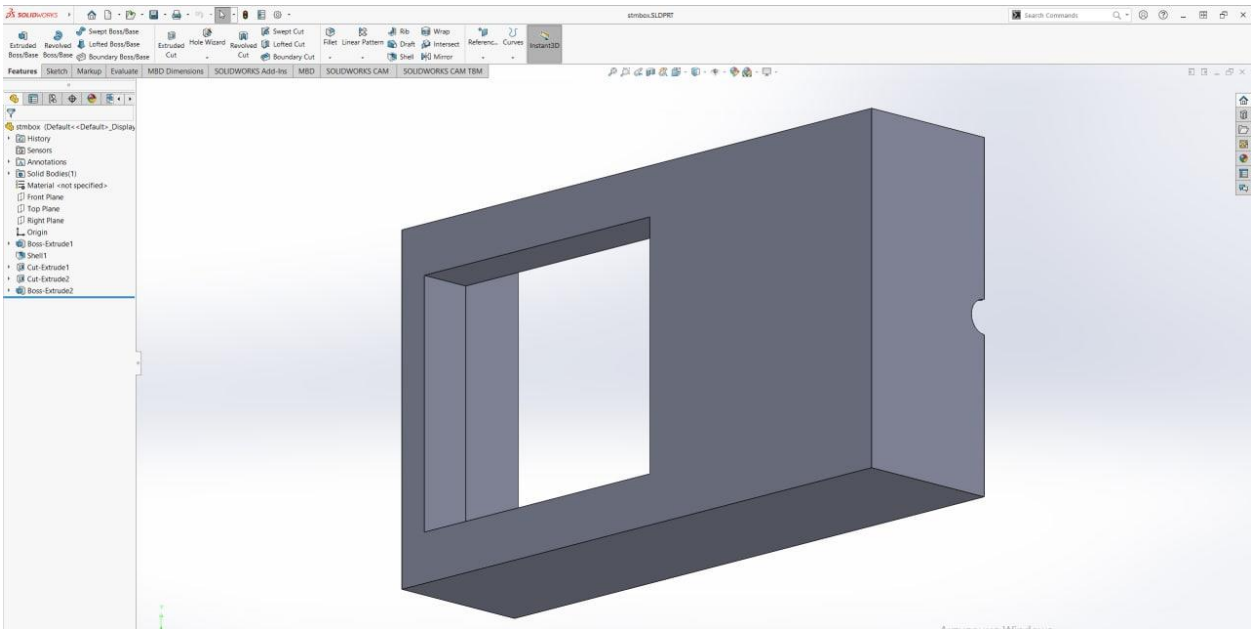

Рисунок 3.5 – 3D модель корпусу керуючого пульту

Зібраний керуючій пульт можливо побачити на рисунку 3.6.

Рисунок 3.6 – Керуючій пульт

Керуючій пульт побудований на базі контролеру STM32F407 та має графічний сенсорний екран для відображення інформації та прийняття команд від користувача. Інтерфейс керуючого пульта можливо побачити на рисунку 3.7.

Рисунок 3.7 – Інтерфейс керуючого пульта

Надрукований на 3D принтері та зібраний стенд представлений на рисунку 3.8.

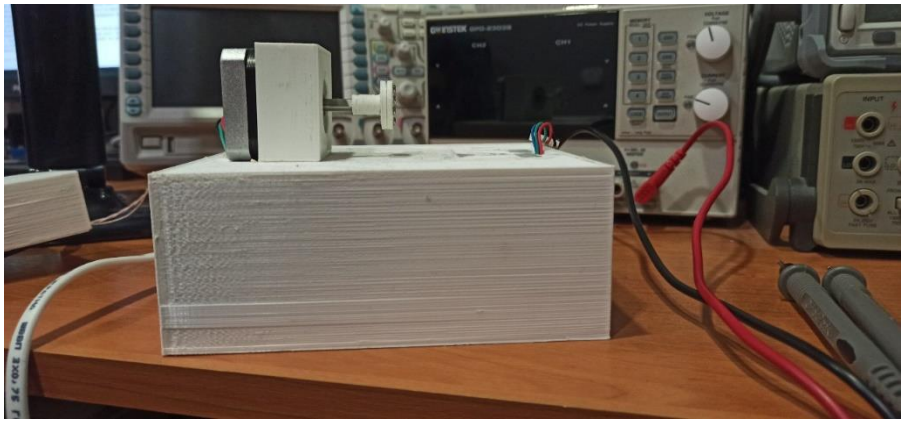

Рисунок 3.8 – Стенд вид збоку

Зібраний стенд з видом зверху представлений на рисунку 3.9.

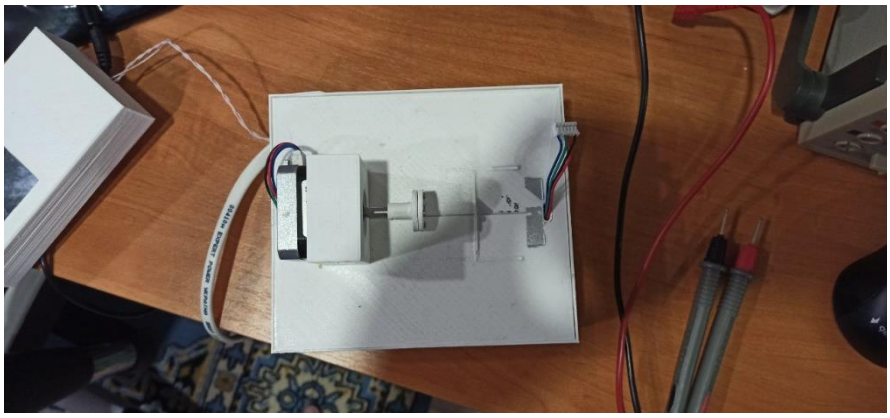

Рисунок 3.9 – Стенд вид зверху

У складі стенду представлені два крокових двигуна (один з них виконує роль навантаження), контролер крокових двигунів, адаптер змінного струму, тестова плата STM32F4 Discovery, DC-DC перетворювач. Мета розробки стенда - можливість імітування умов роботи на підприємстві. Конструкція стенду покликана імітувати роботу насосного агрегату. Приклад насосного агрегату представлений на рисунку 3.10.

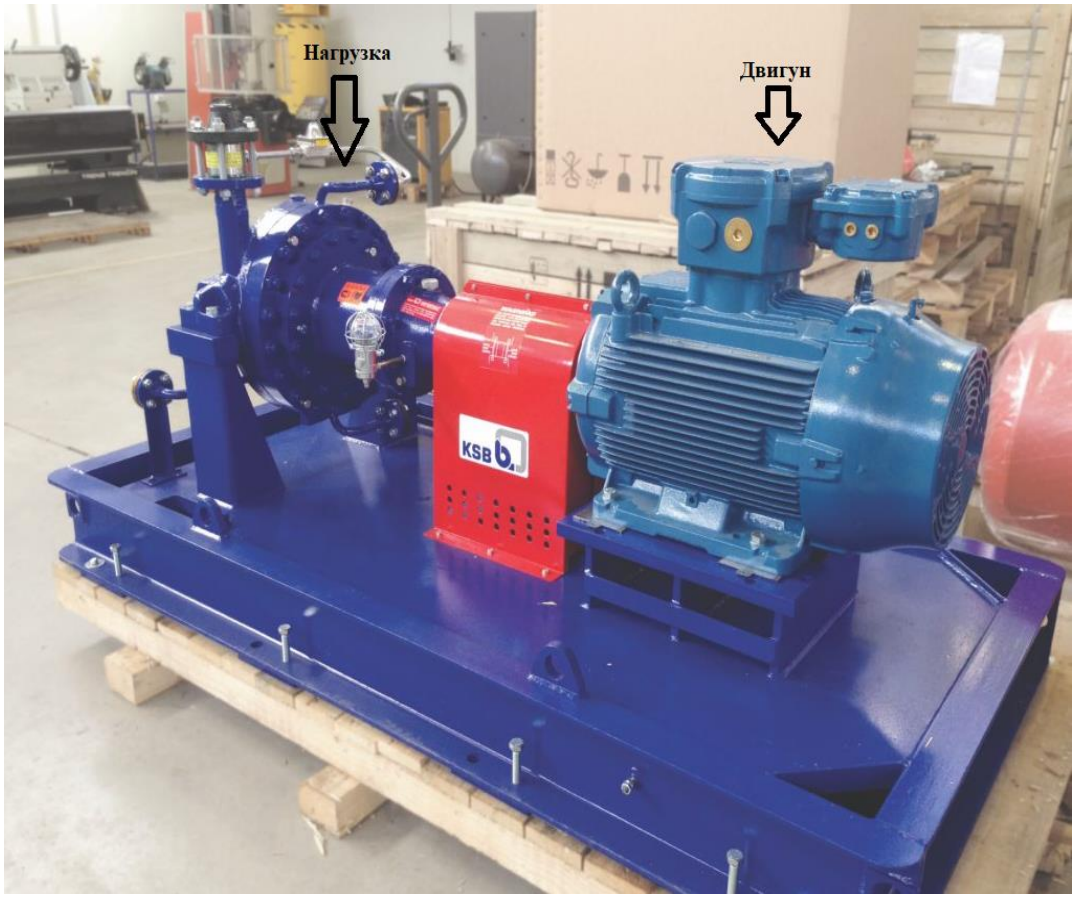

Рисунок 3.10 – Промисловий насосний агрегат

Контроль вібрації в подібних агрегатах проводитися на вузлах навантаження та двигуна. Схематичне уявлення місць кріплення датчиків представлено на рисунку 3.11:

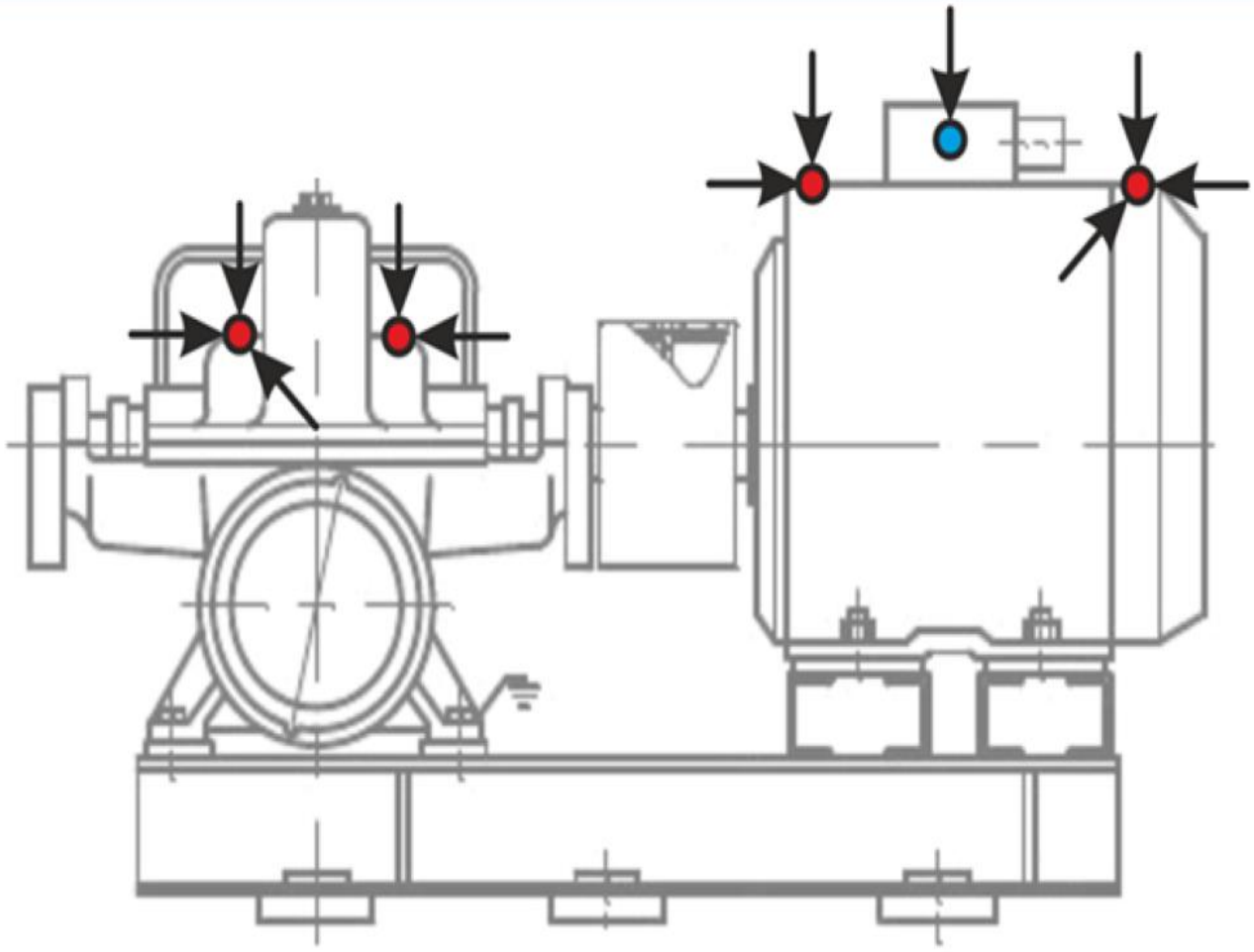

Рисунок 3.11 – точки контролю вібрацій

Одним з переваг обраного складу стенду є можливість регулювати і отримувати показники оборотів не тільки на вбудованому контролері, але і в системі. Це необхідно для зіставлення показників оборотів і отриманих значень датчика вібрацій.

3.1 Оцінка функціонування спроектованого комплексу

Оцінку функціонування доцільно почати з перевірки функціональності калібрування датчика. Для цього можливо використовувати відладки при використанні інтерфейсу JTAG. У датчика три осі і з цієї причини доцільно проводити перевірку калібрування шляхом установки окремої осі перпендикулярно площині робочого столу. Якщо кут витриманий вірно, показання для осі в такому випадку повинні становити значення сили тяжіння землі. Схема розташування осей зображена на рисунку 3.12:

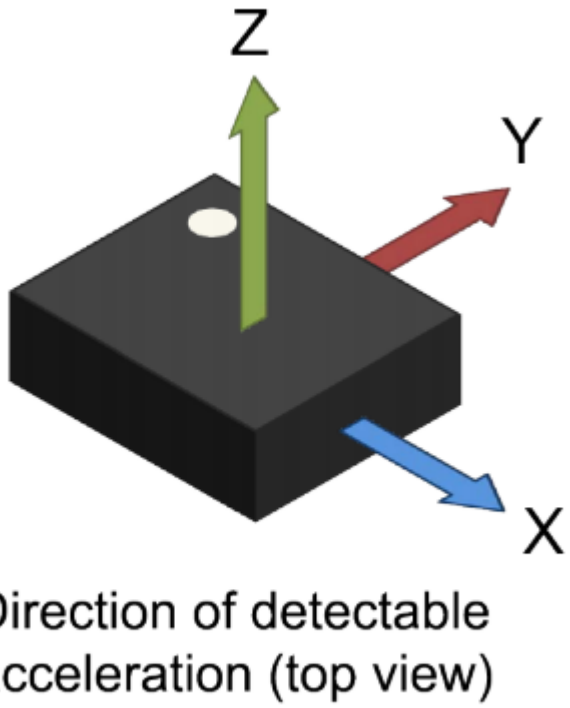

Рисунок 3.12 – Схематичне представлення направлень вимірювання

Результат вимірювань представлено на рисунку 3.13.

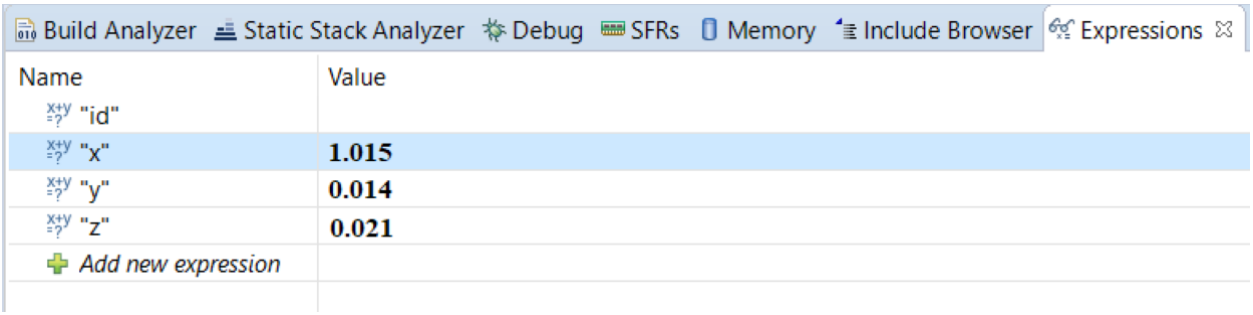

Рисунок 3.13 –Результат вимірювання

Аналогічні результати отримані з іншими осями. Після отримання результатів потрібно перевірити функціональність акселерометра за допомогою розробленого стенду.

Для моделювання оберти стенду виставлені на 190 оборотів в секунду. Результати отримані за допомогою відладчика і візуалізовано засобами відладчика середовища розробки. Результати представлено на рисунку 3.14:

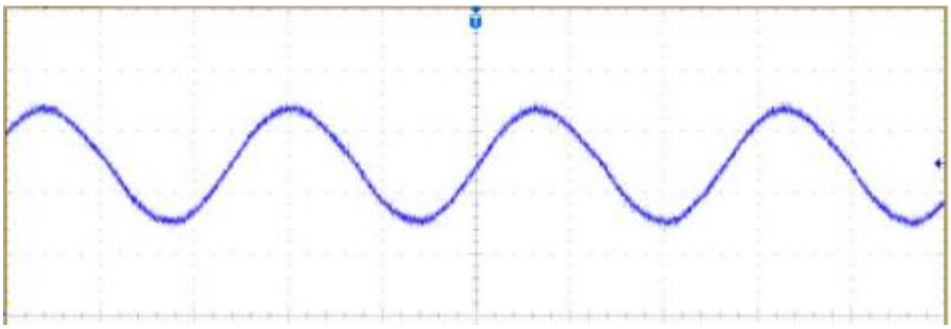

Рисунок 3.14 –Результат випробування стенду за допомогою відладчика

Для перевірки коректності результатів необхідно скопіювати отримані значення в ексель і провести перетворення в частотну область. Результат приведення представлений на рисунку 3.15:

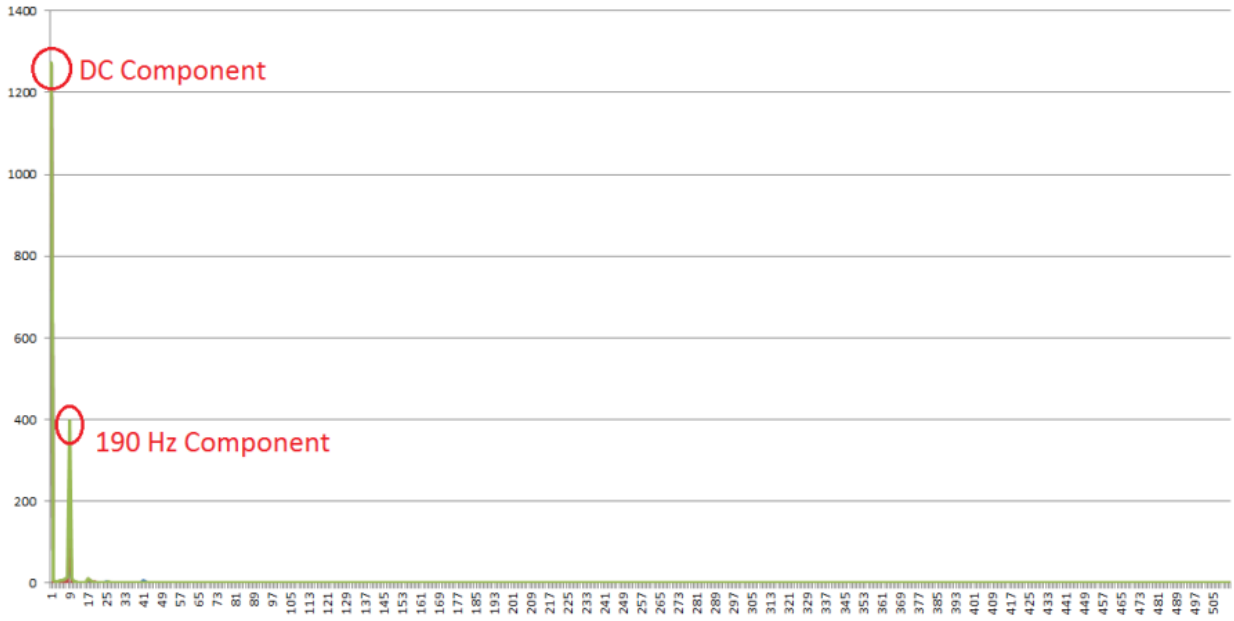

Рисунок 3.15 –Результат випробування стенду за допомогою відладчика після ШПФ

Для проведення експерименту було реалізовано додаток для Android, завдяки якому смартфон може працювати в якості HUB шлюзу.

Інтерфейс дуже простий, функціонал програми дозволяє лише підписатися на повідомлення джерела даних. Вибір проводитися за допомогою GUID?.

Інтерфейс представлений на рисунку 3.16:

Рисунок 3.16 – Інтерфейс реалізації Hub рівня

Після вибору пристрою смартфон упакує повідомлення від BLE і передає в хмарний сервіс. Далі показники будуть візуалізовано за допомогою реалізованості програмного забезпечення. Результат візуалізації представлений на рисунку 3.17:

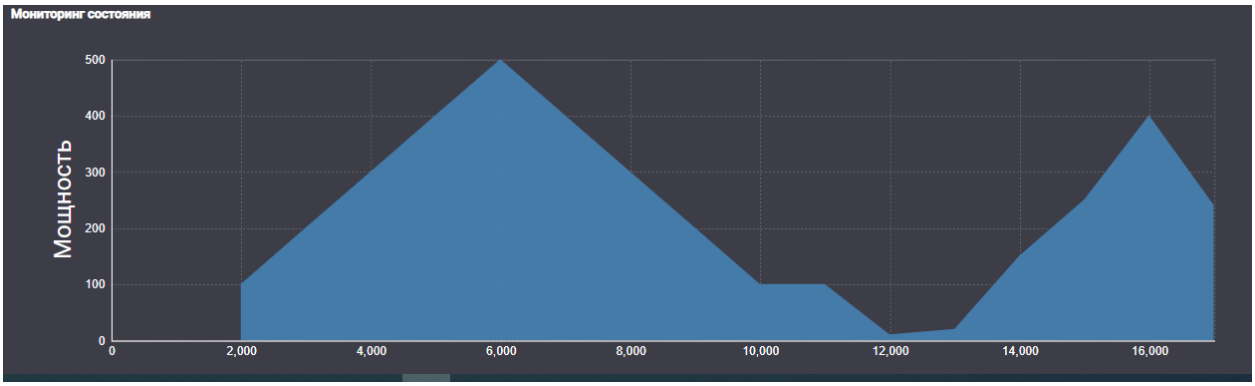

Рисунок 3.17 – Результати візуалізації спектру за допомогою хмарного рішення.

Показники зберігаються та їх можливо переглянути за допомогою спектрограми. Спектрограма з історичними даними представлена на рисунку 3.18.

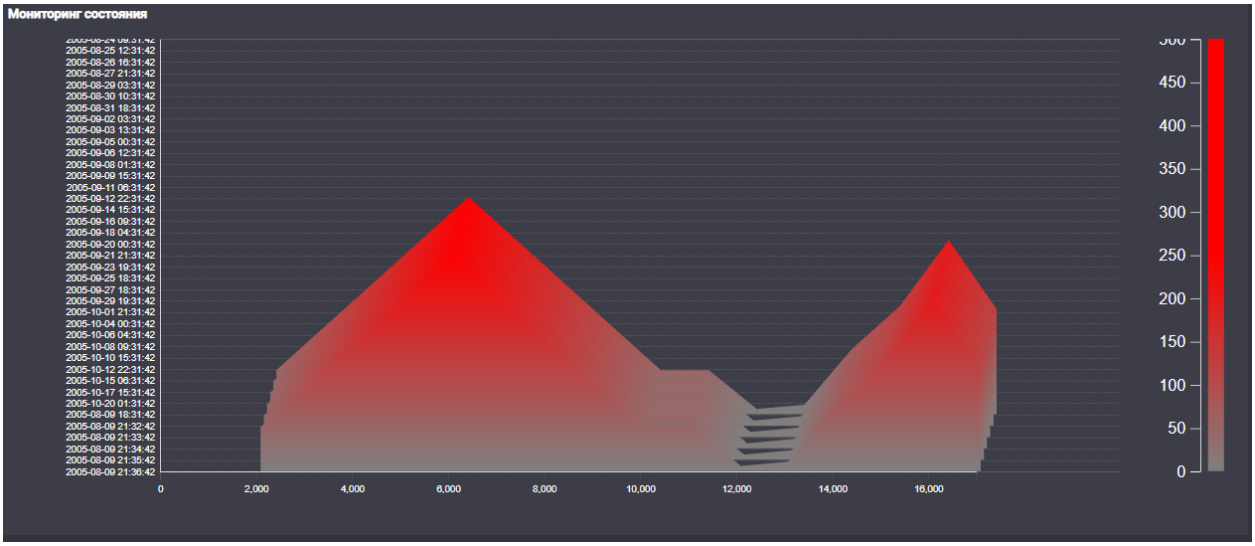

Рисунок 3.18 – Спектрограма з історичними даними

3.2 Практичні рекомендації по використанню системи вібраційної діагностики

Використовуючи спектрограму експерт може приймати рішення про необхідність детальної перевірки обладнання або проведення ТО. Показники які відображає рисунок 3.17 можливо побачити у таблиці 3.1.

Таблиця 3.1 – Показники відображені на рисунку 3.17

Частота, Гц	Вібраційне прискорення. мG
2000	100
6000	500
10000	100
11000	100
12000	10
13000	30
14000	150
15000	250
16000	400

Згідно з показниками в таблиці 3.1 експерт може зробити висновок що вібраційне прискорення 500 мG на частоті 6000 Гц дуже високе і потрібно провести технічне обслуговування.

Для оцінки змін у показників у часі експерт може використовувати спектрограму. Показники які відображає спектрограма на рисунку 3.18 можливо побачити у таблиці 3.2.

Таблиця 3.2 – Показники відображені на рисунку 3.17

Дата та час	Вібраційні показники	
	Частота, Гц	Вібраційне прискорення. мG
2020-10-10 15:22:31	2000	100
	6000	500
	10000	100
	11000	100
	12000	10
	13000	30
	14000	150
	15000	250
	16000	400
2020-10-15 06:31:42	2000	100
	6000	500
	10000	100
	11000	100
	12000	10
	13000	30
	14000	150
	15000	250
	16000	400
2020-10-17 15:31:42	2000	100
	6000	500
	10000	100
	11000	100
	12000	10
	13000	30
	14000	150
	15000	250
	16000	400

За допомогою даних в таблиці 3.2 експерт може зробити висновок що обладнання працює стабільно та не має прогресуючого пошкодження.

Для проведення автоматизованої класифікації створено алгоритм класифікації стану обладнання на базі Dictionary Learning. Алгоритм розгорнутий як окремий сервіс отримує показники від серверу класифікує стан обладнання та видає попередження користувачу при виявленні відхилень.

3.3 Висновки за третім розділом

В ході аналізу результатів експерименту було оцінено роботу системи вібраційної діагностики та сформовані практичні рекомендації по використанню системи. Були виділені наступні переваги системи:

1) відображення показників системи у реальному часі;
2) можливість оцінювати стан обладнання автоматизовано;
3) можливість експертної оцінки стану обладнання за допомогою інструментів системи;
4) проста інсталяція системи як в хмарному сервісі так і на комп’ютері користувача;
5) клієнтський веб-додаток який надає простий доступ з будь якого пристрою.

ВИСНОВКИ

В результаті виконання роботи були виконанні наступні завдання:

1) проведено оцінку методів стану обладнання та аналіз наявних технічних рішень діагностики обладнання, обран оптимальний метод оцінки стану обладнання з використанням вібраційної діагностики;
2) розроблена система вібраційної діагностики та оцінки стану обладнання яка дозволяє контролювати показники вібрації обладнання у реальному часі;
3) виконано планування експерименту для дослідження системи вібраційної діагностики обладнання, створений стенд для імітації реального обертового обладнання;
4) була виконана оцінка придатності до використання системи вібраційної діагностики та визначені її переваги;
5) сформовані практичні рекомендації застосування розроблених методів та програмно технічних рішень для оцінки обладнання за допомогою вібраційної діагностики.

В рамках виконання вищенаведених завдань спроектовано та реалізовано фізичну модель системи, яка включає мікроконтролер, трьохосьовий цифровий акселерометр IIS3DWB та характеризується низькою ціною технічного рішення.

Розроблено та реалізовано спеціалізоване ПЗ системи, яке включає драйвер для налаштування, збору і опрацювання даних з акселерометра та відповідне ПЗ для побудови графіків сигналів віброприскорення в часовій і частотній областях. Побудоване ПЗ дає змогу реалізувати широкі функціональні можливості та є вільновикористовуваним.

Побудована система дає можливість проводити аналіз параметрів вібрації з метою передбачення і запобігання можливих аварій, зменшуючи таким чином затрати пов’язані з виходом із ладу ріжучого інструмента, дорогих деталей і вузлів обертового обладнання.

ПЕРЕЛІК ПОСИЛАНЬ

1. Adams, R.D., Cawley, P., Pye, C.J. and Stone, B.J. (1978). A vibration technique for non-destructively assessing the integrity of structures. Journal of Mechanical Engineering Science, 20(2), 93–100.

2. Banks, H.T., Inman, D.J., Leo, D.J. and Wang, Y. (1996). An experimentally validated damage detection theory in smart structures. Journal of Sound and Vibration, 191(5), 859–880

3. Kessler, S.S., Spearing, S.M., Atalla, M.J., Cesnik, C.E.S. and Soutis, C. (2002). Damage detection in composite materials using frequency response methods. Composites Part B: Engineering, 33, 87–95.

4. Khan, A.Z., Stanbridge, A.B. and Ewins, D.J. (2000). Detecting damage in vibrating structures with a scanning LDV. Optics and Lasers in Engineering, 32, 583–592.

5. Kosmatka, J.B. and Ricles, J.M. (1999). Damage detection in structures by modal vibration characterization. Journal of Structural Engineering, 125(12), 1384–1392.

6. Visual Studio. Посилання: https://visualstudio.microsoft.com/ru/

7. MVC . Посилання: https://dotnet.microsoft.com/apps/aspnet/mvc

8. Особливості ASP.NET.MVC. Посилання: https://metanit.com/sharp/mvc5/1.1.php

9. Генетичний алгоритм Посилання: https://studfiles.net/preview/5465767/

10. Применение генетичсеких алгоритмов Посилання: http://www.machinelearning.ru/wiki/index.php

11. Genetic_algorithm. Посилання: https://en.wikipedia.org/wiki/Genetic_algoritm.html

ДОДАТОК А
Програмний код мобільного додатку

А.1 Лістинг коду SensorValueAggregateService

```
using IoTMonitoring.Core.Entities.AverageValueAggregate;
using IoTMonitoring.Core.Entities.SensorValueAggregate;
using IoTMonitoring.Core.Interfaces;
using System;
using System.Collections.Generic;
using System.Text;
using System.Threading.Tasks;
using System.Net.Http;
using System.Net.Http.Json;
using Newtonsoft.Json;
using System.Linq;

namespace IoTMonitoring.Data.Services
{
	public class SensorValueAggregateService : ISensorValueAggregateService
	{
		public class ModelResult
		{
			public bool Sucsess { get; set; }
			public string Result { get; set; }
		}
		public async Task<AverageValue> GetAverageValue(SensorValue sensorValue)
		{
			AverageValue result = null;
			var content = new { columns = new string[] { "Type", "Data" }, data = new List<string[]> {
				new string[] {
					"predict", JsonConvert.SerializeObject(sensorValue.FreqValues.Select(x
=> new { Frequency = x.Key, Power = x.Value }))
				}
			};
			var stringContent = new StringContent(JsonConvert.SerializeObject(content),
Encoding.UTF8, "application/json");
			stringContent.Headers.ContentType.CharSet = string.Empty;
			using (HttpClient client = new HttpClient()){
				using (HttpResponseMessage res = await client.SendAsync(new
HttpRequestMessage
				{
					Method = HttpMethod.Post,
					RequestUri = new Uri($"http://88d73a10-cf57-4219-8dec-
3cd0969bddbe.westeurope.azurecontainer.io/score"),
					Content = stringContent
				}))
				{
					result = new AverageValue();
					var stringResult = await res.Content.ReadAsStringAsync();
					var modelResult =
JsonConvert.DeserializeObject<ModelResult>(stringResult.Substring(1, stringResult.Length - 2));
					var str = modelResult.Result.Split(',');
					var state = Core.Entities.SensorState.Normal;
					switch(str[0])
					{
						case "Warning":
```

```
                                        state = Core.Entities.SensorState.Warning;
                                        break;
                                case "Error":
                                        state = Core.Entities.SensorState.Error;
                                        break;
                                }

                                result.SensorId = sensorValue.SensorId;
                                result.Date = sensorValue.Date;
                                result.Value = double.Parse(str[1]);
                                result.State = state;
                        };
                        return result;
                }
        }
        public async Task<bool> SetAverageValue(SensorValue sensorValue)
        {
                bool result = false;
                var content = new
                {
                        columns = new string[] { "Type", "Data" },
                        data = new List<string[]> {
                        new string[] {
                                "set", JsonConvert.SerializeObject(sensorValue.FreqValues.Select(x =>
new { Frequency = x.Key, Power = x.Value }))
                        }
                }
                };
                var stringContent = new StringContent(JsonConvert.SerializeObject(content),
Encoding.UTF8, "application/json");
                stringContent.Headers.ContentType.CharSet = string.Empty;
                using (HttpClient client = new HttpClient())
                {
                        using (HttpResponseMessage res = await client.SendAsync(new
HttpRequestMessage
                        {
                                Method = HttpMethod.Post,
                                RequestUri = new Uri($"http://88d73a10-cf57-4219-8dec-
3cd0969bddbe.westeurope.azurecontainer.io/score"),
                                Content = stringContent
                        }))
                        {
                                result = true;
                        };
                        return result;
                }
        }
    }
}
```

А.2 Лістинг коду EfRepository

```
using EdgePipeline.Core.Entities;
using EdgePipeline.Core.Interfaces;
using IoTMonitoring.Data.Data.Things;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace EdgePipeline.Data.Data
{
	public class EfRepository<T> : IAsyncRepository<T> where T : BaseEntity, IAggregateRoot
	{
		protected readonly ThingsContext _dbContext;

		public EfRepository(ThingsContext dbContext)
		{
			_dbContext = dbContext;
		}

		public async Task<int> CountAsync(ISpecification<T> spec)
		{
			return await ApplySpecification(spec).CountAsync();
		}

		public async Task<T> AddAsync(T entity)
		{
			_dbContext.Set<T>().Add(entity);
			await _dbContext.SaveChangesAsync();

			return entity;
		}

		public async Task UpdateAsync(T entity)
		{
			_dbContext.Entry(entity).State = EntityState.Modified;
			await _dbContext.SaveChangesAsync();
		}

		public async Task DeleteAsync(T entity)
		{
			_dbContext.Set<T>().Remove(entity);
			await _dbContext.SaveChangesAsync();
		}

		private IQueryable<T> ApplySpecification(ISpecification<T> spec)
		{
			return SpecificationEvaluator<T>.GetQuery(_dbContext.Set<T>().AsQueryable(), spec);
		}
	}
}
```

А.3 Лістинг коду AnaliticComponent

```
import { HostListener, Directive,SimpleChanges, Component, OnInit, Input, OnChanges, ViewChild, ElementRef,
AfterViewInit } from '@angular/core';
import * as d3 from 'd3';
import { Sensor } from 'src/app/core/entity/Sensor';
import { SensorDataModel } from 'src/app/core/entity/SensorDataModel';
import { SensorsService } from 'src/app/core/services/sensor/sensor.service';
import { FrequencyModel } from 'src/app/core/entity/FrequencyModel';
import {formatDate} from '@angular/common';
@Component({
  selector: 'analitic-graph',
  templateUrl: './Analitic.component.html',
  styleUrls: ['./Analitic.component.css']
})
export class AnaliticComponent implements OnInit, OnChanges {
  @Input() transitionTime = 1000;
  @Input() hticks = 60;
  @Input() data: Sensor;
  @Input() showLabel = 1;
  hostElement; // Native element hosting the SVG container
  svg; // Top level SVG element
  g; // SVG Group element
  colorScale; // D3 color provider
  xAxis;
  yAxis;
  zAxis;
  brush;
  xAxisNet;
  idleTimeout;
  extent;
  startIndex;
  endIndex;
  line;
  clipped;
  maxZ;
  data_area;
  yAxisLegend;
  clip;
  legendScale;
  zoom;
  yAxisNet;
  zAxisNet;
  x; // X-axis graphical coordinates
  y; // Y-axis graphical coordinates
  z;
  overlap = 16;
  drag;
  dates;
  colors = d3.scaleOrdinal(d3.schemeCategory10);
  bins; // Array of frequency distributions - one for each area chaer
  paths; // Path elements for each area chart
  areaGenerator;
  area; // For D3 area function
  histogram; // For D3 histogram function

  constructor(private elRef: ElementRef, public sensorService: SensorsService) {
     this.hostElement = this.elRef.nativeElement;
     this.sensorService = sensorService;
  }
```

```
ngOnInit() {
}

ngOnChanges(changes: SimpleChanges) {
   if (changes.data && !changes.data.firstChange) {
      this.data = changes.data.currentValue
      this.createChart();
   }
}

private createChart() {
   this.removeExistingChartFromParent();
   this.dates = this.data.sensorValues.map(x => new Date(x.date))
   this.startIndex = 0;
   this.endIndex = 39;
   this.dates = this.dates.sort((n1,n2) => {
    if (n1 > n2) {
       return 1;
    }
    if (n1 < n2) {
       return -1;
    }
    return 0;
   })

   this.data.sensorValues.forEach(element => {
    let tmp = []
    element.sensorsFrequencyDataItems = element.sensorsFrequencyDataItems.concat(tmp).sort((n1,n2) => {
       if (n1.frequency > n2.frequency) {
          return 1;
       }
       if (n1.frequency < n2.frequency) {
          return -1;
       }
       return 0;
      });
   });
   this.data.sensorValues = this.data.sensorValues.sort((n1,n2) => {
    if (n1.date < n2.date) {
       return 1;
    }
    if (n1.date > n2.date) {
       return -1;
    }
    return 0;
   })
   if((this.dates.length/60) < 1)
   {
      var count = 60 - this.dates.length
      var minDate = this.dates[0]
      for(var i = 0; i < count; i++)
      {
       var sensDataModel = new SensorDataModel()
       sensDataModel.date = new Date(minDate.setHours(minDate.getHours() + i))
       sensDataModel.sensorsFrequencyDataItems = [];
       this.data.sensorValues.push(sensDataModel)
      }
   }

   this.setChartDimensions();

   this.addGraphicsElement();
```

```
    this.createXAxis();

    this.createYAxis();
    this.createZ();

    this.areaGenerator = d3.area()
        .x((datum: any) => this.x(datum.frequency))
        .y0(270)
        .y1((datum: any) => this.z(datum.power));
    this.line = this.areaGenerator.lineY1();
    this.createAreaCharts();
    this.setZoom();
    this.setColorScale();
    //this.setBrushTool();

  }
  private setColorScale()
  {
   this.g.append("linearGradient")
   .attr("id", "line-gradient")
   .attr("gradientUnits", "userSpaceOnUse")
   .attr("x1", 0)
   .attr("y1", this.z(0))
   .attr("x2", 0)
   .attr("y2", this.z(this.maxZ))
   .selectAll("stop")
     .data([
       {offset: "0%", color: "grey"},
       {offset: "100%", color: "red"}
     ])
   .enter().append("stop")
     .attr("offset", function(d) { return d.offset; })
     .attr("stop-color", function(d) { return d.color; });

  }
  private setZoom()
  {
   this.zoom = d3.zoom()
   .scaleExtent([-Infinity, Infinity])
   .extent([[70, 0], [540, 270]])
   .translateExtent([[70, -Infinity], [540, Infinity]])
   .on("zoom", this.zoomed.bind(this));
   this.svg.call(this.zoom)
   .transition()
     .duration(750)
     .call(this.zoom.scaleTo, 4, [this.y('2005-08-09T18:31:42'), 0]);
  }
  private zoomed() {
   if(d3.event && d3.event.sourceEvent && d3.event.sourceEvent.type == 'wheel')
   {
       var direction = d3.event.sourceEvent.wheelDelta < 0 ? 'down' : 'up';

       if(direction == 'up')
       {
             if(this.startIndex > 0)
             {
                this.startIndex--;
                this.endIndex--;
                this.y = d3.scalePoint().domain(this.data.sensorValues.slice(this.startIndex, this.endIndex).map(x =>
formatDate(x.date, 'yyyy-MM-dd HH:mm:ss', 'en-US'))).range([270, 0])
```

```
            }
        }
        else
        {
            if(this.endIndex < this.data.sensorValues.length)
            {
               this.startIndex++;
               this.endIndex++;
               this.y = d3.scalePoint().domain(this.data.sensorValues.slice(this.startIndex, this.endIndex).map(x =>
formatDate(x.date, 'yyyy-MM-dd HH:mm:ss', 'en-US'))).range([270, 0])
            }
        }
        var current = this.data.sensorValues.slice(this.startIndex, this.endIndex)[0]
        this.sensorService.SetCurrentValueSensor(this.data.id, current.sensorsFrequencyDataItems)
        this.yAxis.transition().duration(this.transitionTime).call(d3.axisLeft(this.y).tickSize(0).tickFormat(<any>''));
        this.yAxisNet.transition().duration(this.transitionTime).call(d3.axisLeft(this.y).ticks(5).tickSize(-500));
        this.createAreaCharts();
    }
  }
  private setChartDimensions() {
     let viewBoxHeight = 300;
     let viewBoxWidth = 750;
     this.svg = d3.select(this.hostElement).append('svg')
        .attr('width', '100%')
        .attr('height', '100%')
        .attr('viewBox', '0 0 ' + viewBoxWidth + ' ' + viewBoxHeight);
  }

  private addGraphicsElement() {
     this.g = this.svg.append("g")
        .attr("transform", "translate(0,0)");

  }


  private createXAxis() {
     this.x = d3.scaleLinear()
        .domain([0, d3.max(this.data.sensorValues, sensorValues => d3.max(sensorValues.sensorsFrequencyDataItems,
dataPoint => dataPoint.frequency))])
        .range([70, 580]);

     this.xAxis =this.g.append('g')
        .attr('transform', 'translate(0,270)')
        .attr("stroke-width", 0.5)

     this.xAxis.call(d3.axisBottom(this.x).tickSize(0).tickFormat(<any>''));

     this.xAxisNet = this.g.append('g')
        .attr('transform', 'translate(0,270)')
        .style('font-size', '6')
        .style("stroke-dasharray", ("1,1"))
        .attr("stroke-width", 0.1)

     this.xAxisNet.call(d3.axisBottom(this.x));


  }

  private createZ() {
     this.maxZ = d3.max(this.data.sensorValues, sensorValues => d3.max(sensorValues.sensorsFrequencyDataItems,
dataPoint => dataPoint.power));
     this.z = d3.scaleLinear()
```

```
        .domain([0, this.maxZ])
        .range([270, 135]);
  }

   private createYAxis() {
     this.y = d3.scalePoint()
        .domain(this.data.sensorValues.slice(this.startIndex, this.endIndex).map(date => formatDate(date.date, 'yyyy-
MM-dd HH:mm:ss', 'en-US')))
        .range([270, 0]);


     this.yAxis = this.g.append('g')
        .attr('transform', 'translate(70,0)')
        .attr("stroke-width", 0.5)

     this.yAxis.call(d3.axisLeft(this.y).tickSize(0).tickFormat(<any>''));

     this.yAxisNet = this.g.append('g')
        .attr('transform', 'translate(70,0)')
        .style("stroke-dasharray", ("1,1"))
        .attr("stroke-width", 0.1)

     this.yAxisNet.call(d3.axisLeft(this.y).ticks(-5).tickSize(-600))
        .style('font-size', '6');

  }

  private createAreaCharts() {
    if(this.paths)
    {
       this.paths.forEach(element => {
          element.remove()
       });
    }
    this.legendScale = d3.scaleLinear()
    .domain([0, this.maxZ])
    .range([270, 0]);
    this.yAxisLegend = this.g.append('g')
    .attr('transform', 'translate(720,0)')
    .attr("stroke-width", 0.5)

    this.yAxisLegend.call(d3.axisLeft(this.legendScale));

    this.g.append("rect")
    .attr("width", 10)
    .attr("height", 270)
    .attr("transform", 'translate(725,0)')
    .style("fill", "url(#line-gradient)");
    this.paths = [];
    this.data.sensorValues.slice(this.startIndex, this.endIndex).reverse().forEach((element,index) => {

       this.paths.push(this.g.append('path')
       .attr('fill', "none")
       .attr("stroke-width", 1)
       .attr("stroke", "url(#line-gradient)" )
       .attr("transform", (d, i) => `translate(${-(index*2)+79},${this.y(formatDate(element.date, 'yyyy-MM-dd
HH:mm:ss', 'en-US'))-270})`)
       .attr("d", this.line(element.sensorsFrequencyDataItems))
       .attr("clip-path", "url(#clip)")
       );
       this.paths.push(this.g.append('path')
       .attr('fill', "url(#line-gradient)")
```

```
      .attr("stroke-width", 1)
      .attr("transform", (d, i) => `translate(${-(index*2)+79},${this.y(formatDate(element.date, 'yyyy-MM-dd
HH:mm:ss', 'en-US'))-270})`)
      .attr("d", this.areaGenerator(element.sensorsFrequencyDataItems))
      .attr("clip-path", "url(#clip)"));
   });
   var current = this.data.sensorValues.slice(this.startIndex, this.endIndex)[0]
   this.sensorService.SetCurrentValueSensor(this.data.id, current.sensorsFrequencyDataItems)
  }
  public updateChart() {
     if (!this.svg) {
        this.createChart();
        return;
     }

     this.x = d3.scaleLinear()
        .domain([0, d3.max(this.data.sensorValues, sensorValues => d3.max(sensorValues.sensorsFrequencyDataItems,
dataPoint => dataPoint.frequency))])
        .range([70, 580]);
     this.xAxis.transition().duration(this.transitionTime).call(d3.axisBottom(this.x).tickSize(0).tickFormat(<any>"));
     this.xAxisNet.transition().duration(this.transitionTime).call(d3.axisBottom(this.x).ticks(10).tickSize(-260));

     this.y = d3.scalePoint()
        .domain(this.data.sensorValues.slice(this.startIndex, this.endIndex).map(date => formatDate(date.date, 'yyyy-
MM-dd HH:mm:ss', 'en-US')))
        .range([270, 0]);
     this.yAxis.transition().duration(this.transitionTime).call(d3.axisLeft(this.y).tickSize(0).tickFormat(<any>"));
     this.yAxisNet.transition().duration(this.transitionTime).call(d3.axisLeft(this.y).ticks(-5).tickSize(-500))
     .style('font-size', '6');

     this.maxZ = d3.max(this.data.sensorValues, sensorValues => d3.max(sensorValues.sensorsFrequencyDataItems,
dataPoint => dataPoint.power));
     this.z = d3.scaleLinear()
        .domain([0, this.maxZ])
        .range([270, 135]);
     this.createAreaCharts();
  }

  private updateAreaCharts() {
     this.paths.forEach((path,index) => {
        path.datum(this.data.sensorValues.slice(this.startIndex, this.endIndex)[index])
           .transition().duration(this.transitionTime)
           .attr("clip-path", "url(#clip)")
           .attr('d', d3.area()
              .x((datum: any) => this.x(datum.frequency))
              .y0(0)
              .y1((datum: any) => this.x(datum.power)));
     });
  }

  private removeExistingChartFromParent() {
     d3.select(this.hostElement).select('svg').remove();
  }
}
```