МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

До захисту допускається
Т.в.о.завідувача кафедри
_____ Сафонова С.О.
«____»_____ 20__ р.

МАГІСТЕРСЬКА РОБОТА

НА ТЕМУ:

«Дослідження роботи та моделювання бездротових сенсорних мереж у контексті IoT»

Освітньо-кваліфікаційний рівень "Магістр"
Спеціальність 123 - "Комп'ютерна інженерія"

Науковий керівник роботи:	_____ (підпис)	Недзельський Д.О. (ініціали, прізвище)
Консультант з охорони праці:	_____ (підпис)	Критська Я.О (ініціали, прізвище)
Студент:	_____ (підпис)	Михайлова А.О. (ініціали, прізвище)
Група:		КІ-18дм

Сєвєродонецьк 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет Інформаційних технологій та електроніки
Кафедра Комп'ютерних наук та інженерії
Освітньо-кваліфікаційний рівень магістр
Напрям підготовки _____
(шифр і назва)
Спеціальність _____
(шифр і назва)

ЗАТВЕРДЖУЮ:
Т. в.о. завідувача
кафедри _____
_____ С.О. Сафонова
«_____» _____20____р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Михайлова Аліса Олександрівна
(прізвище, ім'я, по батькові)

1. Тема роботи «Дослідження роботи та моделювання бездротових сенсорних мереж у контексті IoT»

керівник проекту (роботи) Недзельський Д.О., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від " " 2019 р. № ____
2. Строк подання студентом роботи _____
3. Вихідні дані до роботи Матеріали дипломної практики

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1) Аналіз методів і засобів моделювання бездротових сенсорних мереж. Постановка задачі досліджень.
2) Дослідження внутрішньої страуктури, стандартів та протоколів бездротових сенсорних мереж.
3) Моделювання БСМ та налаштування моделей з використанням протоколів маршрутизації.
4) Охорона праці та безпека в надзвичайних ситуаціях.

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

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Недзельський Д.О., доцент		
2	Недзельський Д.О., доцент		
3	Недзельський Д.О., доцент		
4	Критська Я.О., ст. викл.		

7. Дата видачі завдання _____

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Формулювання теми проекту	3.09.19 – 5.09.19	
2	Аналіз предметної області	6.09.19 – 15.09.19	
3	Аналіз методів моделювання БСМ	16.09.19 – 26.09.19	
4	Аналіз програмних засобів	27.09.19 – 8.10.19	
5	Формулювання ТЗ	9.10.19	
6	Дослідження структури БСМ	11.10.19 – 25.10.19	
7	Дослідження стандартів та протоколів	26.10.19 – 13.11.19	
8	Підготовча робота з ПЗ	14.11.19 – 18.11.19	
9	Створенння моделі мережі	19.11.19 – 25.11.19	
10	Налаштування моделі	26.11.19 – 18.12.19	
11	Проведення експерименту	19.12.19 – 25.12.19	
12	Порівняльний аналіз результатів	26.12.19 – 29.12.19	
13	Оформлення пояснювальної записки	30.12.19 – 6.01.20	
14	Розробка презентації	7.05.20	
15	Розробка автореферату	8.05.20 – 10.05.20	

Студент _____ Михайлова А.О.
(підпис) (прізвище та ініціали)

Науковий керівник _____ Недзельський Д.О.
(підпис) (прізвище та ініціали)

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

БСМ, WSN (wireless sensor network)	– бездротова сенсорна мережа.
IoT (internet of things)	– інтернет речей.
ПЗ	– програмне забезпечення
LEACH (Low Energy Adaptive Clustering Hierarchy)	– самоорганізуючий, адаптивний протокол кластеризації, який використовує рандомізацію для рівномірного розподілу енергетичного навантаження між датчиками в мережі та використовується, як простий протокол маршрутизації в бездротових сенсорних мережах (БСМ).
TDMA (Time Division Multiple Access)	– спосіб використання радіочастот, коли в одному частотному інтервалі знаходяться кілька абонентів, різні абоненти використовують різні тимчасові слоти (інтервали) для передачі
TEEN (Threshold sensitive Energy Efficient sensor Network protocol)	– протокол маршрутизації реактивної кластеризації.
HEED (Hybrid Energy Efficient Distributed clustering)	– протокол, який використовується для кластеризації і приносить енергоефективну маршрутизацію кластеризації з чітким врахуванням енергії.
CH (cluster head)	– голова кластера.
BS (base station)	– базова станція.

ВСТУП

Обґрунтування вибору теми дослідження

Інноваційні технології останніх років нерозривно пов'язані з IoT або ж Інтернетом Речей, який дозволяє зменшити потребу в безперервній участі людини в процесі обробки інформації. Однією з основ подібних інновацій є бездротові сенсорні мережі (БСМ). Особливістю даних мереж можна назвати їх структуру, яка являє собою самоорганізовану мережу безлічі датчиків і виконавчих пристроїв, об'єднаних між собою за допомогою радіоканалу. БСМ використовуються для вирішення багатьох питань, таких як [1]:

- Своєчасне виявлення можливих відмов виконавчих механізмів, з контролю таких параметрів, як вібрація, температура, тиск і т. п .
- Контроль доступу до віддалених систем об'єкта моніторингу в режимі реального часу;
- Автоматизація інспекції та технічного обслуговування промислових активів.
- Керування комерційними активами.
- Застосування як компоненти в енерго- і ресурсозберігаючих технологіях.
- Контроль екологічних параметрів навколишнього середовища.

Слід зазначити, що незважаючи на тривалу історію сенсорних мереж, концепція побудови сенсорної мережі остаточно не оформилася і не висловилася в певні програмно-апаратні (платформні) рішення. Саме тому ця тема потребує подальшого ретельного дослідження, аналізу, моделювання, тестування та порівняння з більш вивченими галузями мережевого проектування. Крім того, питання необхідності завчасного моделювання подібних мереж є дуже важливим у контексті уникнення появи проблем на стадіях розробки та експлуатації.

O.V. Tuzhilkin, N.S. Ulianin у своїй науковій статті «METHODS OF EVALUATING THE PERFORMANCE OF WIRELESS SENSOR NETWORK» досліджують методології подальшого обчислення ефективності БСМ та їх моделей з використанням методу критеріїв [2].

Тому обґрунтованою є тема магістерської роботи, у якій вирішується **науково-прикладне завдання** розроблення моделей і методу інформаційної технології моделювання бездротових сенсорних мереж.

Об'єкт дослідження – процеси забезпечення ефективного моделювання бездротових сенсорних мереж, їх дослідження та зв’язку з IoT.

Предмет дослідження – моделі та метод інформаційної технології оцінки і забезпечення ефективного моделювання БСМ з урахуванням особливостей їх структури.

Мета і завдання дослідження. Метою дослідження є підвищення ефективності моделювання БСМ з використанням спеціалізованого ПЗ та математичних моделей.

Для досягнення мети дослідження необхідно вирішити такі **завдання**:

– аналіз методів і засобів моделювання БСМ;

– розроблення моделей БСМ;

– практичний та математичний аналіз отриманих моделей БСМ.

Методи дослідження. Проведені в роботі дослідження основані на методах моделювання, які використовувались при розробці бездротової сенсорної мережі. Базуються на використанні відповідного ПЗ (OMNeT++), порівнянні метрик у моделях та підрахунку ефективності роботи моделей за критерієм сумарної середньої затримки.

Особистий внесок здобувача полягає у розробленні нових моделей, методів та інструментальних засобів, що дозволяють вирішити поставлені задачі. Усі основні результати отримані автором особисто. У роботах, опублікованих у співавторстві, автору належать: конференція «Майбутній науковець» 2018 (тези «Налаштування мережі з використанням динамічного протоколу маршрутизації OSPF») [3], конференція «TACSIT-2019» (стаття «Analysis and modeling of dynamic routing network protocols») [4].

Апробація матеріалів дисертації. Основні положення, ідеї, висновки магістерської роботи доповідалися та обговорювалися на конференції TACSIT-2019.

Зв'язок з науковими програмами, планами, темами. Магістерська робота виконана у Східноукраїнському національному університеті ім. В. Даля у відповідності з державними програмами і планами НДР: Internet of Things: Emerging Curriculum for Industry and Human Applications (ALIOT) (Реєстрац. № 573818-EPP-1-2016-1- UK-EPPKA2-CBHE-JP).

Практичне значення отриманих результатів полягає в тому, що отримані результати дають розширене розуміння принципів роботи БСМ, їх налаштування, протоколів та особливостей моделювання. Отримана відповідь на питання про доцільність використання програмних засобів для моделювання БСМ.

Публікації. За темою магістерської роботи з викладенням її основних результатів опубліковано 2 наукових праць, серед яких 1 статей у наукових фахових виданнях України; 1 тези доповідей міжнародних конференцій.

РОЗДІЛ 1
АНАЛІЗ МЕТОДІВ І ЗАСОБІВ МОДЕЛЮВАННЯ БЕЗДРОТОВИХ СЕНСОРНИХ МЕРЕЖ. ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕНЬ

1.1 Аналіз вимог

Об'єктом дослідження є бездротові сенсорні мережі та методи їх моделювання.

Новизна та недостатня вивченість галузі бездротових сенсорних мереж, а також стрімкий розвиток IoT, призводять до необхідності проведення комплексних досліджень. Особливо гостро стоїть питання фінансування подібних мереж та апаратів, що робить попереднє моделювання необхідним кроком для створення правильно функціонуючої мережі з мінімальніми затратами на етапі проектування. Моделювання БСМ здійснюється за допомогою як математичних моделей, так і відповідного програмного забезпечення.

У роботі вирішується питання більш глибокого дослідження БСМ, вибору програмного забезпечення для їх моделювання та перевірки ефективності створеної моделі шляхом аналізу отриманих результатів. Важливою частиною роботи є порівняльний аналіз протоколів маршрутизації, які були використані при моделюванні.

Далі надана основна інформація з теми дослідження БСМ.

1.1.1 Основні поняття про БСМ

Новітні технології та прогрес в області проектування та подальшого виробництва мікросхем дозволили перейти до впровадження нового класу комунікаційних систем – бездротових сенсорних мереж [1].

БСМ (wireless sensor networks) у своєму складі мають мініатюрні обчислювально-комунікаційні пристрої - моти (від англ. Motes - пилинки), або сенсори. Мот являє собою об'єкт розміром не більше одного кубічного дюйма.

Ці вузли (моти) є недорогими, малопотужними та багатофункціональними пристроями для виконання різних завдань зондування. Сенсорні вузли розгорнуті по всій області для моніторингу певних подій (наприклад, температури) в реальних умовах. БСМ в основному працюють у відкритій і некерованій зоні. Передбачається, що вони будуть грати важливу роль в різних галузях, наприклад, військове спостереження, моніторинг лісових пожеж, моніторинг безпеки будівель і керування виробничими процесами.

Кожен вузол сенсорної мережі зазвичай містить порти вводу/виводу даних з різних датчиків контролю зовнішнього середовища (або самі датчики), мікроконтролер і радіо-

приймач, а також автономне або зовнішнє джерело живлення [1]. Це дозволяє пристрою отримувати результати вимірювань, проводити початкову обробку даних, і підтримувати зв'язок із зовнішньою інформаційною системою. Мікроконтролер може бути використаний для реалізації інтелектуальної розподіленої обробки даних. В інтелектуальній бездротовій сенсорній мережі пристрої здатні на локальному рівні обмінюватися інформацією, аналізувати її та передавати до певної глибини оброблену інформацію, а не "сирі" дані. Це дозволяє значно скоротити вимоги до пропускної здатності мережі, збільшити масштабованість і термін експлуатації системи. Однак додавання "інтелекту" в мережу вимагає врахування особливостей прикладної задачі, тому цей підхід, як правило, ефективний при розробці вузькоспеціалізованої системи.

Таким чином ключовими особливостями сенсорних мереж є:

- здатність самоорганізації мережі передачі інформації та її адаптація до чисельного складу пристроїв;
- здатність ретрансляції повідомлень між елементами;
- можливість наявності датчиків в кожному елементі;
- тривалий термін автономної роботи (1 рік і більше) [1].

Одним з перших прообразів сенсорної мережі можна вважати систему СОСУС, призначену для виявлення та ідентифікації підводних човнів. В середині 1990-х років технології бездротових сенсорних мереж стали активно розвиватися, на початку 2000-х років розвиток мікроелектроніки дозволив виробляти для таких пристроїв досить дешеву елементну базу.

Бездротові мережі початку 2010-х років в основному базуються на стандарті ZigBee.

1.1.2 Стандарт IEE 802.15.4

В експлуатаційному плані основними відмітними особливостями БСМ є вимоги стійкого функціонування в умовах динамічних змін в топології мережі через переміщення сенсорів, автономне електроживлення і суттєві обмеження в енергоспоживанні та обчислювальної продуктивності вбудованих в вузли мережі мікроконтролерів, пам'яті, трансиверів та інших мікроелектронних компонентів [5]. При цьому в той же час умови функціонування БСМ передбачають передачу выдносно малих обсягів інформації з малою швидкістю. З огляду на запити ринку телекомунікацій в специфічній сфері моніторингу та керування об'єктами за допомогою бездротового зв'язку під керывництвом IEEE (Institute of Electrical and Electronics Engineers) році була випущена офіційна специфікація IEEE 802.15.4, що стала стандартом. За планами розробників, новий стандарт повинен був забезпечити дальність з'єднання, порівнянну з WiFi, але при цьому мати менше енергоспоживання за рахунок низької швидкості передачі

даних. В ряду найважливіших завдань також забезпечення роботи в режимі реального часу з використанням тимчасових слотів, запобігання колізій доступу і комплексна підтримка захисту мереж. Сумісні зі стандартом 802.15.4 пристрої повинні мати можливість керування витратою електроенергії та контролю якості з'єднань. З травня 2007 року сертифіковані пристрої 802.15.4, потужність випромінювання яких не перевищує 10 мВт на відкритій місцевості і 100 мВт в приміщенні [5].

У документі 802.15.4 визначені два нижніх рівні моделі OSI: канальний та фізичний. Фізичний рівень керує способом передачі даних, апаратними особливостями та параметрами, які необхідні для побудови мережі. На практиці фізичний рівень визначає роботу трансивера, виконує вибір каналів, та рівень потужності передачі.

У відповідності зі специфікацією стандарту 802.15.4 на фізичному рівні під обмін даними зарезервовані 27 каналів в трьох частотних діапазонах: 868 МГц, 910 МГц, 2.4 ГГц, що дозволяє використовувати стандарт в неліцензованих в більшості країн світу частотних смугах.

IEEE 802.15.4 робить можливою двосторонню напівдуплексну передачу даних, підтримуючи при цьому шифрування AES 128. Доступ до каналу містить у собі принцип CSMA/CA. CSMA/CA - це один з мережевих протоколів, яуий використовує принцип прослуховування несучої частоти. Пристрій, який готовий до передачі даних посилає jam signal сигнал затору і прослуховує ефір. Якщо буде виявлено "чужий" сигнал затору, то передавач "засинає" на випадковий проміжок часу, а потім знову пробує почати передачу кадру. Таким чином, передачу може виконувати тільки один пристрій, що підвищує продуктивність мережі [6]. При цьому дані передаються відносно невеликими пакетами, що характерно для трафіку сигналів керування і моніторингу в БСМ. Важливою особливістю стандарту є обов'язкове підтвердження доставки повідомлень.

Особливістю пристроїв, об'єднаних в мережу по стандарту IEEE 802.15.4, є низьке енергоспоживання за рахунок переходу трансивера в режим "засипання" при відсутності даних для пересилання і збереження підключення в цьому режимі. При розробці стандарту, основний акцент робився на швидкість процесів конфігурації та реконфігування. Зокрема, перехід приймача в активний стан триває близько 10-15 мс, а підключення до основної мережі додаткових пристроїв - від 30 мс. При цьому тривалість реконфігурації та підключення пристроїв залежить від сталості "прослуховування" маршрутизаторами мережі.

Типи вузлів мережі. Стандарт визначає два типи вузлів мережі: повнофункціональний пристрій FFD (Fully Function Device), який може реалізувати як функцію координації роботи і установки параметрів мережі, так і працювати в режимі типового вузла; пристрій з обмеженим набором функцій RFD (Reduced Function Device), що може тільки підтримувати зв'язок з повнофункціональними пристроями. У будь-якій мережі повинен бути, принаймні, один FFD,

який реалізує функцію координатора [7]. Кожен пристрій має 64-бітний ідентифікатор, але в деяких випадках для обмеженої області може використовуватися короткий 16-бітний для з'єднань в персональній мережі PAN (personal area network).

Стандартом підтримується і більш структурована топологія "зірка", в якій координатор (FFD) мережі обов'язково повинен бути центральним вузлом формованої персональної мережі (PAN) з унікальним ідентифікатором. Після цього інші пристрої можуть приєднуватися до мережі, яка повністю незалежна від інших мереж з аналогічною топологією.

Стандарт 802.15.4 описує два нижніх рівня моделі OSI, не визначаючи вимог до верхніх рівнів і умов їх сумісності. Вирішення цих завдань вимагало розробки спеціальних комунікаційних протоколів. Найбільш відомими є протоколи альянсу ZigBee, який був створений у 2002 році найбільшими світовими компаніями, що спеціалізуються в області розробки програмно-апаратних засобів для інфокомунікаційних систем. У числі більш ніж двохсот членів альянсу ZigBee, які координують роботи з просування технологій та виробництва технічних засобів для бездротових сенсорних мереж - Texas Instruments, Motorola, Philips, IBM, Samsung, LG, OKI та багато інших [7]. Корпорація Intel, хоча не є членом альянсу ZigBee, активно підтримує його діяльність. ZigBee розробив і ратифікував у 2004 році стандарт, що включає повний стек протоколів для бездротових сенсорних мереж [7]. Стандарт ZigBee бере за основу стандарт IEEE 802.15.4, що описує виключно фізичний рівень і рівень доступу до середовища для бездротових мереж передачі даних з низьким енергоспоживанням [7]. На відміну від нього документ ZigBee включає опис мережевих процесів керування, сумісності та профілів пристроїв, а також інформаційної безпеки. На мережевому рівні в ZigBee визначені механізми маршрутизації і формування логічної топології мережі [7].

1.1.3 Сфери використання БСМ

Важливість використання БСМ зростає з кожним роком. Це безпосередньо пов'язано зі збільшенням потреби контролю, спостереження, вимірювань і рішення багатьох інших задач експлуатації в таких областях як промисловість, медицина, комерція, наука, побут.

Найбільш відомі області застосування БСМ:

Військова техніка.

Для застосування у військових цілях потрібен добре оснащений і надійний бездротовий датчик, який може витримати особливі умови експлуатації (наприклад, підвищена температура, вологість та інше) при цьому мати компактний розмір і конструкцію, яка не привертає уваги противника [8]. Особливу увагу у військовій сфері слід приділяти моніторингу появи несправностей для своєчасного їх усунення. Можливість використання бездротових датчиків у

військовій області варіюється від моніторингу транспортних засобів (дружніх або протиборчих), моніторинг можливих загроз і багатьох інших цілей з щільною топологією для збору більш надійних даних.

Медичне устаткування.

В даний час бездротові датчики є затребуваними в медицині для спрощення взаємозв'язку між пацієнтом і системою моніторингу. Також є функції, які виконуються за допомогою медичних датчиків, таких як контроль захворювань і введення препаратів. Для поліпшення дистанційного моніторингу життєво важливих показань пацієнта підвищують чутливість датчика.

Екологічні програми.

БСМ може бути використана для вимірювання декількох параметрів навколишнього середовища, таких як температура, вологість, тиск, інтенсивність світла, і характеристика ґрунту. Вона також використовується для відстеження, контролю за рухом і поведінкою тварин, птахів та інших істот. У більшості випадків сенсорні вузли прикріплені до рухомих істот або щільно розміщені всередині цільового середовища. Деякі функції вимагають контрольованості датчика для йогокерування. Екологічне застосування вимагає тривалої автономної роботи з протоколами передачі даних для спостереження і контролю у важкодоступних місцях проживання об'єкта дослідження.

Побутова техніка.

Активне використання БСМ не залишилось поза увагою людини в повсякденному житті. Керування будинком/оргтехнікою за допомогою пульта дистанційного керування, який дає можливість всередині цільової області змінювати параметри пристроїв шляхом прямого зв'язку між користувачем і пристроями, за допомогою мережі Інтернет або супутникового зв'язку [8]. Для інтерактивності між побутовою технікою і користувачем потрібен штучний інтелект, який за допомогою сенсорних вузлів розвиває свої реакції на адаптується до потреб користувача.

1.1.4 Зв’язок IoT та БСМ

«Інтернет речей» – технологія, що дозволяє об'єднати в мережу за допомогою Інтернету пристрої, що виконують різні цілі та завдання – починаючи від холодильника і телевізора, і закінчуючи, дата-центрами і фермами для майнінгу [8].

Інтернет – речей (IoT), як технологія, своїй появі зобов'язана імплементації технології радіочастотного обміну між пристроями, здатними комунікувати в загальній мережі.

Вперше презентація цієї інноваційної технології була проведена в 1999 році в Массачусетському університеті для фармацевтичної мережі «Procter and Gamble». Основна мета

такої презентації була довести ефективність використання радіочастотних міток в системах складської та торгової логістики. Для такого найбільшого виробника споживчої косметики подібна схема керування логістикою була одним із важелей впливу на конкурентноспособність на ринку. Для IoT була розроблена уніфікована технологія RFID. Базовою платформою комунікації став високошвидкісний інтернет (3G і 4G), за допомогою якого можна комбінувати мережі будь-якої складності, де беруть участь пристрої різної конфігурації та призначення.

Інтернет речей будується на базі розгалуженої мережі різних пристроїв, сенсорних датчиків та інших приладів, що зчитують, фіксують певні фізичні параметри. Спочатку вся концепція IoT будувалася на використанні технології радіочастотної ідентифікації (Radio Frequency Identification) і бездротової сенсорної мережі (БСМ), Ця розподілена мережа передбачає використання одного або декількох радіочастотних каналів. Через канали зв'язку, об'єднані в мережу пристрої, взаємодіють один з одним. Площа покриття таких локальних мереж може становити від декількох десятків метрів до декількох квадратних кілометрів. Основним радіочастотним каналом, який буде використаний в IoT, є діапазон частот від 3 до 3.9 ГГц. Цей діапазон частот обраний не випадково. Він сполучається з частотами, на яких, як передбачається, буде працювати супутниковий сегмент широкосмугового Інтернету (проекти One Web або Sky Link).

Всі пристрої IoT використовують технологію RFID, працюючи за принципом виявлення та ідентифікації спеціальних електронних міток або маркерів (транспондерів). Тобто система в чому-то аналогічна тій, що використовується у військовій авіації – «свій - чужий». Упізнані як свої пристрої об'єднуються в єдину інформаційну мережу і виконують ті чи інші завдання. Наприклад, електронні мітки на продуктах, що встановлюються в магазинах, допомагають в подальшому смарт-холодильникам реалізувати програму керування запасами продовольства в конкретному будинку [8].

1.1.5 Моделювання БСМ

Моделювання - це процес створення моделі як концептуального уявлення деякого явища. Зазвичай модель буде мати справу тільки з деякими аспектами даного явища, і дві моделі одного і того ж явища можуть істотно відрізнятися, тобто відмінності між ними будуть не тільки в простому перейменуванні їх складових компонентів.

Моделі зазвичай використовуються, коли неможливо або непрактично створювати експериментальні умови, при яких вчені можуть безпосередньо вимірювати результати. Незважаючи на те, що прямий експеримент завжди дає більш точні результати, моделювання, як попередній етап розробки, істотно спрощує і здешевлює наступні етапи, мінімізуючи

можливі помилки.

Моделювання дозволяє знизити витрати витрати часу та фінансів на розробку і налагодження бездротових сенсорних мереж.

Основними методами моделювання та перевірки ефективності бездротових сенсорних мереж є:

- використання спеціалізованих програм моделювання.
- аналіз з використанням математичних моделей;

Використання спеціалізованих програм значно спрощує процес моделювання і дозволяє наочно роздивитися основні особливості роботи мережі, а математичні моделі допомагають спрогнозувати поведінку мережі з заданими вхідними умовами.

Не дивлячись на різноманіття допоміжних засобів, конкретних та найефективніших шляхів моделювання бездротових сенсорних мереж не виявлено, через відносну новизну цього питання.

Для вирішення поставленого питання запропоновано провести порівняльний аналіз програмних засобів та підтвердити показники за допомогою математичних моделей.

1.1.6 Безпека у БСМ

Основні цілі інформаційної безпеки в БСМ умовно розподіляють на дві категорі: першочергові та другорядні. Першочергові містять у собі конфіденційність, цілісність, аутентифікацію та доступність данних [11]. Серед другорядних: свіжість даних, самоорганізація, тимчасова синхронізація, захищена локалізація.

Конфіденційність даних є фундаментальним завданням безпеки. Конфіденційність в сенсорних мережах повинна захистити передані по мережі дані з метою закрити до них доступ для потенційних зловмисників за допомогою різних механізмів, таких як контроль доступу, шифрування і т. д.

Аутентифікація даних необхідна для підтвердження достовірності даних за допомогою ідентифікації їх походження/першоджерела. Аутентифікація даних дозволяє перевірити легітимність відправника і одержувача даних в мережі. Аутентифікація даних забезпечується за допомогою застосування симетричних і асиметричних механізмів, де відправляючі і приймаючі вузли сенсорної мережі обмінюються секретними ключами. Завдяки бездротовій передачі даних і особливості роботи сенсорних мереж без постійного залучення людини забезпечення аутентифікації даних стає досить складним завданням.

Цілісність даних в сенсорних мережах визначається здатністю забезпечення захисту даних таким способом, щоб дані не могли бути змінені під час транспортування між вузлами

сенсорної мережі. Наприклад, цілісність даних мережі може перебувати під загрозою в разі наявності в мережі скомпрометованого вузла, який впроваджує в мережу неправдиві дані, в разі втрати або пошкодження даних внаслідок нестабільних умов і бездротової природи сенсорних мереж [11].

Доступність даних має на увазі можливість роботи сенсорної мережі за допомогою транспортування даних між її вузлами. У разі порушення цієї властивості даних, сенсорна мережа не може продовжувати функціонувати і підтримувати виконання покладених на неї функцій. Доступність даних сенсорної мережі може бути легко скомпрометована за допомогою атаки і виведення з ладу базової станції або головного вузла кластера сенсорної мережі.

Свіжість даних в бездротових сенсорних мережах дозволяє визначити, що дані були отримані датчиком і відправлені по мережі в перший раз, а не є копією старих даних повторно відправлених в мережу. З метою забезпечення свіжості даних в пакет даних може бути впроваджений тимчасовий лічильник, який буде служити сигналом до видалення пакета в разі перевищення певного значення.

Самоорганізація вимагає від кожного сенсора мережі бути достатньо незалежним і гнучким для можливості самовідновлення в топології в залежності від різних ситуацій. Дана властивість необхідно тому, що сенсорні мережі зазвичай є децентралізованими за своєю природою, тобто не мають фіксованої інфраструктури. Дана особливість доставляє певні труднощі в забезпеченні безпеки. У разі відсутності самоорганізації збиток, отриманий в результаті атаки, може бути руйнівним.

Тимчасова синхронізація є необхідною властивістю, на якій ґрунтується успішна робота різних механізмів і протоколів в сенсорних мережах. Як і в будь-якій розподіленій системі, тимчасова синхронізація необхідна в сенсорних мережах для визначення загальної шкали часу для всіх вузлів мережі та їх локальних вбудованих часових механізмів.

Захищена локалізація необхідна для забезпечення здатності сенсорної мережі з точністю і автоматично виявляти кожен вузол мережі (наприклад, несправний вузол). На жаль, зловмисник може з легкістю маніпулювати незахищеною інформацією про місцезнаходження, як за допомогою неправдивих повідомлень про інтенсивність сигналу або за допомогою відтворення сигналів.

Більшість загроз у БСМ схожі з погрозами і атаками на провідні мережі, за винятком того що БСМ дещо важче захистити. Рис. 1.1, 1.2 ілюструють класифікацію загальних атак і атак на бездротові сенсорні мережі.

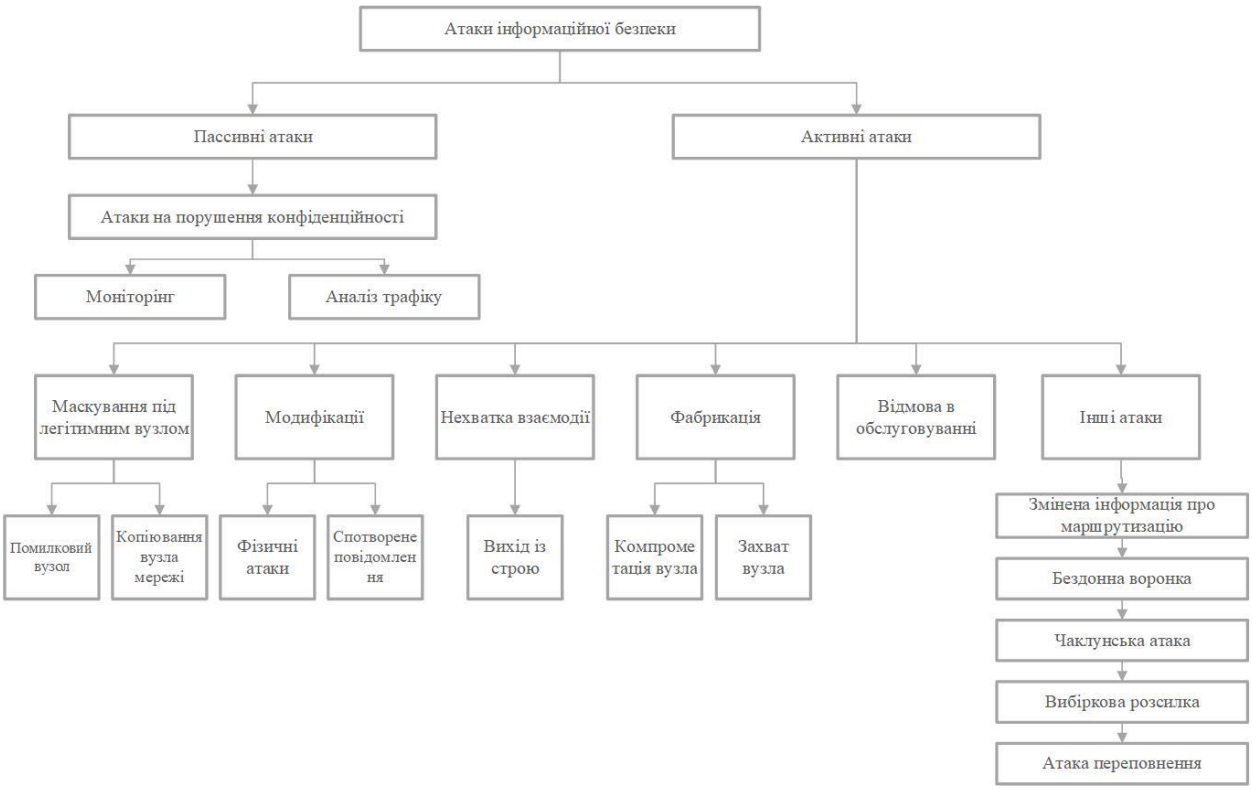


Рисунок 1.1 – Загальна класифікація атак

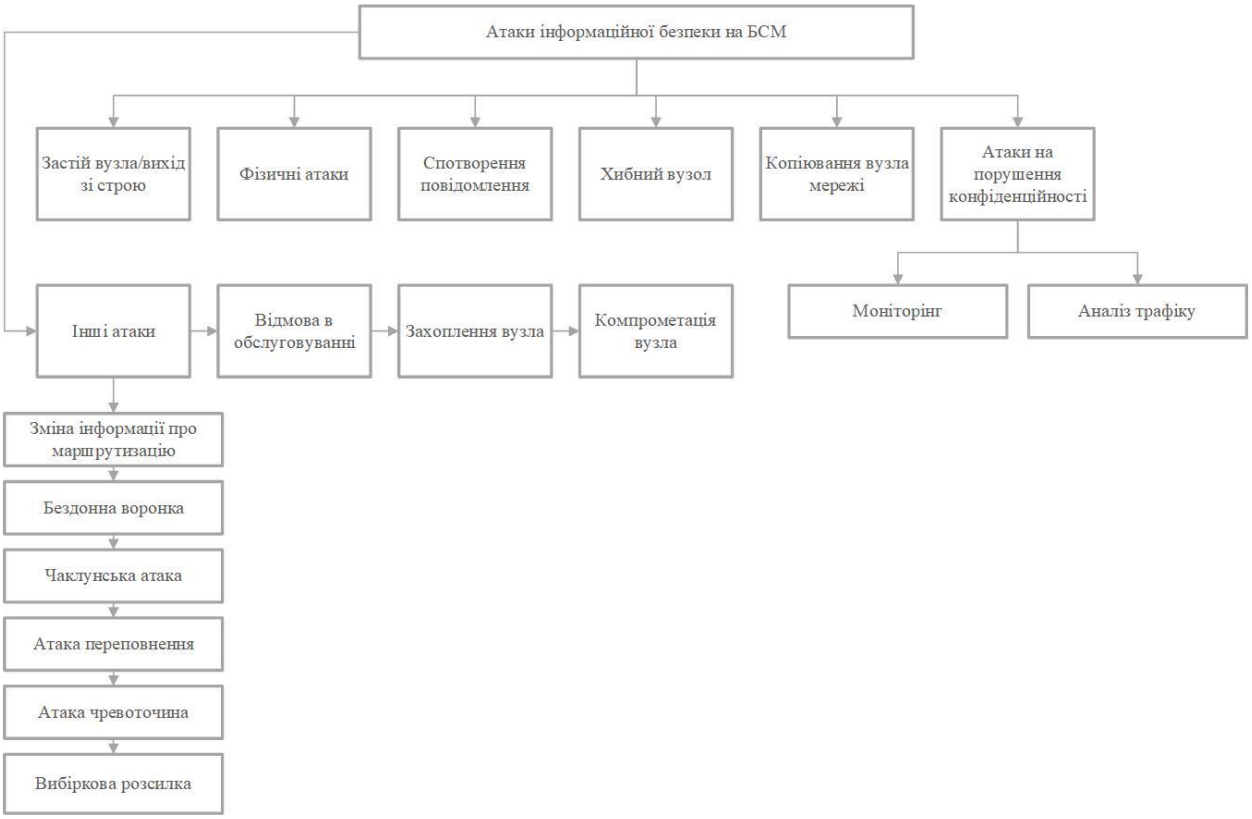


Рисунок 1.2 – Класифікація атак на БСС

Механізми забезпечення безпеки використовуються для ідентифікації, запобігання і

відновлення після атак. Механізми забезпечення безпеки можна умовно розділити на механізми високого і низького рівня. Рис. 1.3 ілюструє класифікацію механізмів безпеки.

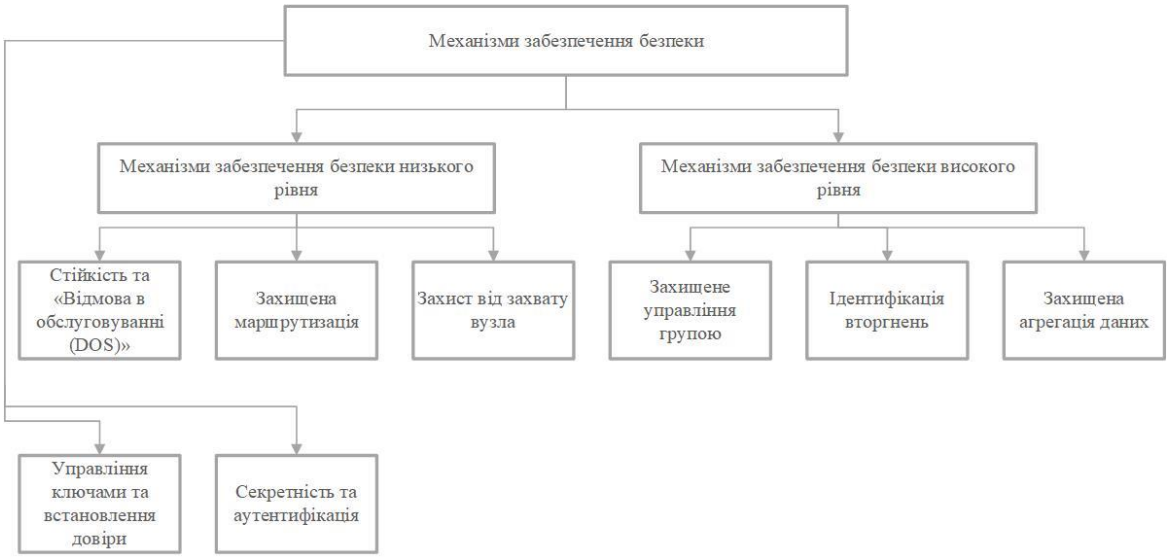


Рисунок 1.3 – Механізми безпеки

Далі наведений короткий опис представлених механізмів:

- *Керування ключами і встановлення довіри*. Через лімітовані ресурси, особливо ресурси енергетичної батареї, асиметричне шифрування ключами не повинно використовуватися для сенсорних мереж. Таким чином, необхідно використовувати симетричне шифрування. Техніки встановлення і керування ключами повинні бути придатними для використання в мережах з сотнями і тисячами вузлів. Крім того, комунікаційні патерни сенсорних мереж відрізняються від традиційних аналогів, вузли сенсорної мережі повинні встановлювати ключі з їх сусідами і з вузлами, що агрегують інформацію. Недолік цього методу полягає в тому, що зловмисники, скомпрометувавши велику кількість вузлів мережі, можуть відновити весь пул ключів і розшифрувати дані.
- *Секретність і аутентифікація*. Сенсорні мережі вимагають захисту від прослуховування, впровадження і модифікації пакетів. Криптографія є стандартним захистом. Складнощі виникають в процесі застосування криптографії для сенсорних мереж. У мережах з рівноправними вузлами, криптографія з межкінцевим шифруванням (end to end cryptography) дозволяє досягти високого рівня безпеки, однак вимагає встановлення ключів між усіма вузлами мережі і є несумісною з широкомовною розсилкою і пасивною участю (технологія, завдяки якій вузол, що прослуховує сусідній до нього вузол мережі, може вирішити не передавати дані, в разі якщо точно такі дані передаються сусіднім вузлом).

- *Стійкість до відмов в обслуговуванні (DOS)*. Причинами відмови в обслуговуванні можуть бути неполадки апаратного забезпечення, помилки в програмному забезпеченні, нестача ресурсів, умови зовнішнього середовища або сукупність впливу перерахованих факторів. Наприклад, зловмисник може спробувати вивести сенсорну мережу з ладу за допомогою передачі потужного сигналу, який здатний повністю заглушити всі комунікації вузлів сенсорної мережі. Технологія розширеного спектру використовується для захисту сенсорних мереж від подібних атак. Вона передбачає використання методів навмисного розширення діапазону частот сигналу. Діапазон частот стає більшим, ніж необхідно для передачі повідомлення. Передача такого сигналу схожа на шум, що дозволяє знизити ризики навмисної інтерференції сигналу з боку зловмисників.
- *Захищена маршрутизація*. Маршрутизація є ключовим процесом, без якого неможливо здійснення комунікації між вузлами сенсорної мережі. Однак сучасні протоколи маршрутизації містять безліч вразливостей інформаційної безпеки. Найпростіші атаки можуть мати на увазі впровадження скомпрометованої маршрутної інформації в сенсорну мережу, що згодом створює проблеми в передачі даних від відправника в кінцеву точку призначення мережі. Розробка нових схем аутентифікації і захищених протоколів маршрутизації може захистити мережі від подібних атак.
- *Захист від захоплення вузла*. Захоплення вузла є серйозною проблемою захисту даних в сенсорних мережах. Часто сенсорні мережі встановлюються в легкодоступних для зловмисників місцях. Зловмисник, захопивши вузол мережі, може дістати криптографічну інформацію, перепрограмувати вузол мережі або замінити вилучений вузол зловмисними вузлами. Найбільш поширеними методами захисту є використання захищеної від злому упаковки, алгоритмічних рішень, техніки хешування.
- *Захищене керування групою*. Кожен вузол бездротової сенсорної мережі обмежений в обчислювальних ресурсах і комунікаційних можливостях. Однак такі функції як агрегація мережевих даних і їх аналіз може здійснювати група вузлів сенсорної мережі. Наприклад, група сенсорів мережі може виконувати спільне стеження за пересуванням певного об'єкта. Вузли сенсорної мережі в групі можуть постійно і швидко змінюватися. Внаслідок цього, необхідні захищені протоколи для керування групами вузлів сенсорної мережі, які повинні дозволяти захищене прийняття вузлів в функціональні групи і підтримувати захищені комунікації вузлів-членів функціональних груп.

- *Ідентифікація вторгнень*. Бездротові сенсорні мережі схильні до різних вторгненнь. Основне завдання механізмів ідентифікації вторгнень полягає в моніторингу сенсорної мережі, ідентифікації можливих спроб проникнення і розсилки відповідних повідомлень користувачам. Для децентралізованих механізмів ідентифікації вторгнень використання захищених груп може бути перспективним підходом.
- *Захищена агрегація даних.* Дані збираються з вузлів сенсорної мережі потім часто агрегуються на рівні базової станції. Завдяки агрегації даних за допомогою сенсорної мережі можна розрахувати середню температуру в географічному регіоні, комбінувати дані сенсорних вузлів для визначення розташування і швидкості транспортного засобу і т. д. Точки агрегації даних схильні до різних видів атак і повинні бути надійно захищені. Скомпрометовані вузли можуть бути використані для впровадження помилкових даних, які згодом призведуть до неправильно агрегованих розрахунків. Захищені протоколи маршрутизації і схеми аутентифікації корисні для запобігання впровадження помилкових даних в сенсорну мережу [11].

1.2 Аналіз програмних та інструментальних засобів

Далі розглянуті основні програмні засоби, які застосовуються для моделювання БСМ та обґрунтовується вибір ПЗ для подальшого виконання роботи.

1.2.1 Симулятор NS-2

NS-2 є програмним забезпеченням (ПЗ) з відкритим кодом (Open Source software), призначеним для дискретно-подієвого моделювання дротяних і бездротових (мобільних) систем зв'язку. Основними мовами в складі симулятора є C++ і TCL (Tool Command Language). Для створення симуляцій використовується OTCL (Object TCL). Програма розміщена у вільному доступі, її можна завантажити на сайті програми і використовувати в академічних цілях.

Симулятор підтримує велику кількість протоколів, типів мереж, елементів мережі, моделей передачі даних. Для моделювання ad-hoc мереж підтримуються протоколи маршрутизації AODV, DSDV, DSR і TORA, які вимагають додаткового доопрацювання для забезпечення можливості роботи з мобільними вузлами [12].

З точки зору об'єкта, що знаходиться в імітаційному моделюванні, він представляє собою окремий об'єкт мережевих об'єктів (Network Objects), який може бути визначений у будь-якому випадку і реагувати на деяке велике число подій (Events). Обов'язки всіх об'єднаних мережевих об'єктів називають планувальником (Scheduler), який містить хронологічну таблицю

в часі. Кожне питання має обов'язкові атрибути: час наступних і мережевих об'єктів, до яких він відноситься.

Слід згадати, що в перших версіях моделі були реалізовані базові функції мережевого рівня ZigBee, але пізніше вони були виключені з загального доступу, оскільки не в повній мірі відповідали даним стандарту. У зв'язку з цим на поточний момент можна використовувати тільки існуючі в NS-2 протоколи маршрутизації, які не до кінця враховують особливості бездротових сенсорних мереж.

Документації щодо застосування симулятора досить мало, бракує навчальної літератури. Пропонується звертатися до списку поширених запитань і аналізувати вихідний код моделі.

1.2.2 Симулятор TOSSIM

TinyOS – система, яка спеціально розроблена для сенсорних мереж. Вона має компонентну програмну модель, описану мовою nesC. TinyOS не є операційною системою в традиційному розумінні. Це програмне середовище для вбудованих систем і набірів компонентів, які дозволяють створювати імітаційні моделі конкретним додатком, наприклад, таким як TOSSIM [13].

Симулятор TOSSIM може моделювати мережі розмірністю до декількох тисяч вузлів, і аналізуючи їх, передбачати поведінку мережі з високою точністю. Моделюючи мережі з можливими перешкодами і помилками, симулятор створює просту, але в той же час ефективну модель всіляких взаємодій вузлів в мережі. Описуючи малопотужну модель пристроїв TinyOS, симулятор моделює поведінку сенсорного вузла з великою вірогідністю, описуючи його характеристики і проводячи велику кількість експериментів. Для зручності розробників, TOSSIM підтримує графічний інтерфейс користувача, забезпечуючи детальну візуалізацію і відтворення дій запущеної імітаційної моделі.

Наведемо основні характеристики TOSSIM [13]:

- маштабованість – симулятор підтримує модель мережі, що складається з великої кількості вузлів з різною конфігурацією. Найбільша з усіх розроблених мереж TinyOS складається приблизно з 850 вузлів, симулятор здатний підтримувати такі моделі;
- достовірність – симулятор описує різні взаємодії вузлів, які можуть виникнути в реальній мережі;
- зв'язаність – симулятор пов'язує алгоритм побудови з його графічним представленням, дозволяючи розробникам тестувати програмний код, який вимагає запуску в реальному пристрої, а також виробляти візуалізації мережі.

Архітектура TOSSIM (рис. 1.4) складається з наступних елементів:

- дискретний потік подій;
- набір програмних компонентів, які імітують апаратні компоненти реальних мотів;
- засоби зв'язку, що надають іншим програмам можливість взаємодіяти з TOSSIM.

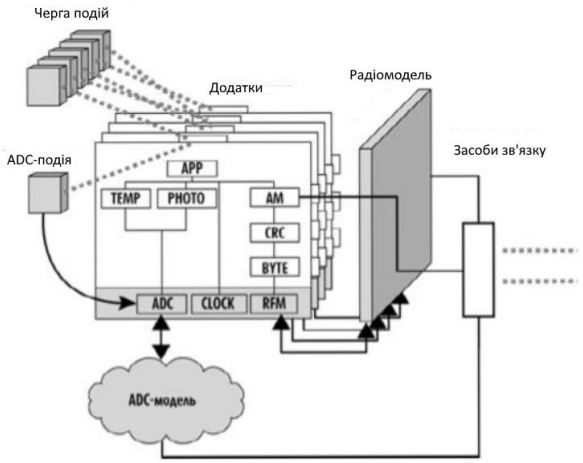


Рисунок 1.4 - Архітектура эмулятора TOSSIM

1.2.3 Симулятор OMNeT++

Даний симулятор являє собою систему моделювання на основі дискретних подій яка може бути використана для таких завдань як:

- моделювання дротових і бездротових комунікаційних систем;
- протоколів моделювання;
- моделювання мереж масового обслуговування.

Програма OMNeT++ підходить для моделювання будь-якої мережі, основою якої є дискретна подія. Процес зручно відображається у вигляді об'єктів, що обмінюються повідомленнями.

OMNeT++ використовує мову C++ для імітаційних моделей. Імітаційні моделі в сукупності з мовою високого рівня NED збираються у великі компоненти і являють собою великі системи. Симулятор має графічні інструменти для створення моделей і оцінки результатів в режимі реального часу.

Моделі програми збираються з компонентів множинного використання, званих модулями. Модулі можна використовувати багато разів і об'єднувати за принципом блоків LEGO [13].

Модулі з'єднуються між собою за допомогою портів, і об'єднуються в складові модулі з використанням високорівневої мови програмування NED. Кількість модулів необмежена.

Модулі зв'язуються за допомогою передачі повідомлень, які містять довільні структури даних. Модулі можуть передавати повідомлення до певних портів і з'єднань сервера або безпосередньо один одному. Останнє, наприклад, корисно для моделювання бездротових мереж.

Процес моделювання може працювати в різних призначених для користувача інтерфейсів. Графічно анімований призначений для користувача інтерфейс зручний для демонстрації та налагодження мережі, а інтерфейс командного рядка зручний для внесення змін.

Компоненти OMNeT++:

1) коренева бібліотека моделювання;

2) OMNeT++ IDE на базі платформи Eclipse;

3) графічний інтерфейс виконуваного моделювання, посилання на виконуваний файл (Tkenv);

4) призначений для користувача інтерфейс командного рядка для виконання моделювання (Cmdenv);

5) документація, приклади.

OMNeT++ працює на базі найпоширеніших операційних систем: (Linux, Mac OS / X, Windows).

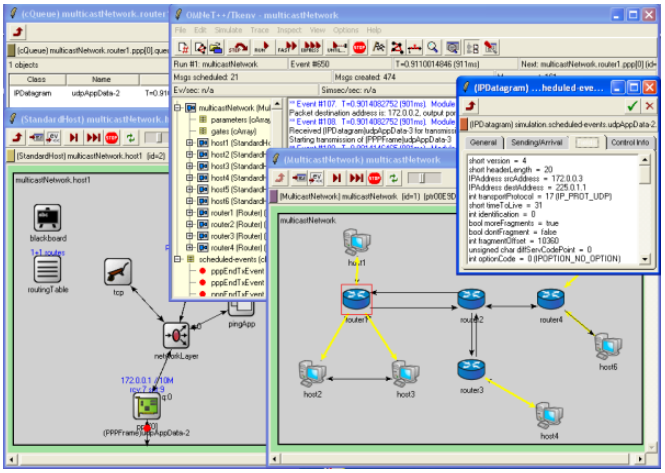


Рисунок 1.5 – Приклад інтерфейсу OMNet++

1.2.4 Порівняльна таблиця програмного забезпечення

Таблиця 1.2 – Порівняння ПЗ за основними параметрами

Назва системи	Графічний інтерфейс	Орієнтованість спеціалізації на БСМ	Можливість побудови БСМ будь-якої складності
NS-2	Ні	Ні	Ні

Продовження таблиці 1.2

TOSSIM	Так	Так	Ні
OMNeT++	Так	Так	Так

1.2.5 Вибір програмного забезпечення

Виходячи з представленої інформації для моделювання було обрано симулятор OMNet++ за наступними параметрами:

1. Можливість промоделювати будь-яку мережу, компоненти якої взаємодіють між собою за допомогою передачі повідомлень.
2. Підтримка основними платформами.
3. Велика кількість допоміжних матеріалів та відео уроків.
4. Підтримка моделювання великих мереж. Обмеження лише за можливостями комп'ютера, який використовується у ході моделювання.
5. Простий інтерфейс зрозумілий на інтуїтивному рівні.
6. Безкоштовне поширення ПЗ.
7. Повна реалізація основних функцій мережевого рівня ZigBee.

1.3 Аналіз математичних моделей

Для оцінки ефективності створеної моделі необхідно провести відповідні дослідження, які включають у себе математичні моделі розрахунку [14].

На сьогоднішній день відсутня єдина методика, яка дозволяє зробити комплексну оцінку ефективності роботи бездротової сенсорної мережі. Найпоширеніший метод визначення ефективності системи ґрунтується на оцінці її роботи по окремо взятому критерію. Такий метод досить простий і дієвий. В даному випадку найбільш ефективною вважається та система, яка вигідно відрізняється від інших за обраним параметром.

Для оцінки якості мереж і їх каналів зазвичай використовуються такі параметри, як пропускна здатність і затримка [15]. Ці параметри дозволяють оцінити ефективність роботи мережі загального призначення. Бездротові мережі зі складною структурою складаються з відносно великої кількості вузлів і каналів зв'язку і при цьому фактично не є мережами загального призначення. Таким чином, ефективність мережі неможливо оцінити лише за допомогою показників якості для одиночного каналу або вузла [2]. Отже, оцінку ефективності з одиночним критерієм необхідно проводити для всієї мережі в цілому. Далі надані три критерії оцінки якості [15], орієнтовані на типові завдання, які вирішуються за допомогою бездротових

сенсорних мереж. Крім того, виявлено метод оцінки з використанням багатьох критеріїв одночасно.

1.3.1 Критерій сумарної середньої затримки

$$\sum_{i=1}^{N} M(h_{iB}) \to \min \tag{1.1}$$

де N – кількість вузлів мережі;

$M(h_{iB})$ – математичне очікування часу доставки даних від вузла і до базової станції. Включає в себе час безпосередньої передачі, а також інші затримки, такі як час ініціалізації передачі канальним рівнем, очікування в чергах вузлів-маршрутизаторів, та інші. Такий критерій підходить для систем, де одинична затримка доставки пакета некритична, однак в цілому від мережі очікується ефективна продуктивність.

Як приклад можна привести системи для наукових досліджень, збору статистичної інформації з метою подальшої обробки, системи обліку для промислових об'єктів. Передбачається, що збір і обробка інформації в таких системах - два різних, слабо пов'язаних за часом етапи [2].

1.3.2 Критерій максимальної затримки

$$\max_{t \in T} (h_{iB}(t)) \to \min \tag{1.2}$$

де T – час роботи системи;

$h_{iB}(t)$ – час доставки даних від вузла і до базової станції в момент часу t. Такий критерій може застосовуватися для системи реального часу, яка вимагає необхідного рівня якості зв'язку для кожного вузла, причому неприпустимо навіть короткочасне погіршення якості зв'язку.

Прикладами таких систем є: військові системи виявлення об'єктів, системи контролю технічного стану промислових агрегатів, системи попередження аварій на промислових об'єктах. Особливістю таких систем є можлива необхідність швидкої реакції у відповідь на зміну параметрів, що реєструються сенсорами. Наприклад, при появі людини в небезпечній зоні, необхідно негайно відключити працюєче там обладнання [2].

1.3.3 Критерій середньої затримки

$$\max_{i=1...N} (M(h_{iB})) \to \min \tag{1.3}$$

Такий критерій можна використовувати для систем реального часу, які вимагають необхідного рівня якості зв'язку для кожного вузла, проте допускають короткочасні збої. Цей критерій підходить, наприклад, для систем, призначення яких носить розважальний характер. До таких систем пред'являються високі вимоги, однак тимчасові погіршення якості зв'язку не призводять до серйозних наслідків [2].

1.3.4 Використання декількох критеріїв

Такі методи оцінки ефективності гарні лише в тому випадку, коли необхідно вибрати систему, яка оптимальна по одному з вибраних параметрів при заданих умовах функціонування. Але що ж робити, коли ефективність роботи системи потрібно оцінити за двома і більше параметрами? При цьому всі параметри, за якими ведеться оцінка, повинні вносити певну вагу в підсумкову оцінку і найбільш повно відображати ефективність роботи при обраних режимах функціонування.

Складні системи характеризуються багатьма параметрами, причому найчастіше у одних системах близькі до оптимуму одні параметри, у інших - інші. В таких умовах непросто оцінити системи без спеціального формалізованого правила критерію. Запропонована методика передбачає проведення оцінки ефективності по довільній кількості параметрів. Оцінка ефективності проводиться в два етапи. На першому етапі з системою проводяться різні тести, за допомогою спеціально розроблених моделей. Тести дозволяють визначити значення окремих параметрів - критеріїв оцінки. Тести повинні проводитися при максимально можливих несприятливих умовах функціонування системи: високий рівень шуму в радіоефірі, наявність перешкод, вихід з ладу частини елементів мережі та ін. Кожен з тестів дозволяє оцінити один параметр системи при заданих умовах роботи. На другому етапі оцінки роботи системи пропонується використовувати вагові критерії ефективності. Ваговий критерій дозволяє врахувати значимість приватних критеріїв, отриманих на першому етапі оцінки.

Значення інтегрального критерію ефективності визначається виразом:

$$k = \sum_{i=1}^{N} a_i k_i \tag{1.4}$$

де N – число приватних критеріїв, k_i – їх значення для оцінюваної системи, а a_i – їх вагові коефіцієнти.

Найкраща система обирається за максимальним значенням критерію:

$$k(s) = \max_{s \in S} \sum_{i=1}^{N} a_i \frac{k_i}{k_{i\max}} \tag{1.5}$$

Вагові коефіцієнти беруться позитивними для максімізованих і негативними для мінімізованих приватних критеріїв. Для вибору величин вагових коефіцієнтів застосовується два підходи:

1. Як коефіцієнти використовують дробові числа, сума яких для кожної системи дорівнює одиниці,

2. Використовують цілі коефіцієнти, причому для найменш важливого приватного критерію беруть одиничний коефіцієнт, а для інших беруть коефіцієнти, кратні одиниці [2].

1.4 Постановка наукової задачі та обґрунтування методики досліджень

Результати проведеного аналізу моделей, методів й інструментальних засобів дослідження та моделювання бездротових сенсорних мереж, показали, що у відомих публікаціях не вирішеною є задача вибору основного методу моделювання бездротових сенсорних мереж та порівняння роботи окремих протоколів маршрутизації.

У відомій літературі задачі, пов’язані з моделюванням бездротових сенсорних мереж не розглядалися у повному обсязі. Не розв’язувалися також задачі з пошуку найбільш придатних для моделювання програмних засобів та дослідження особливостей побудови моделей з їх викорстанням. Даних щодо використовуємих протоколів маршрутизації та порівняння її роботи також критично мало. Крім того, виходячи з відносної новизни галузі, майже відсутні структуровані дослідницькі роботи з даної теми, особливо на території України. Недостатньо висвітлений зв’язок зі сферою IoT.

В даному контексті слід визначити 3 задачі магістерської роботи:

1) пошук методу моделювання, який би дозволив поліпшити процес моделювання БСМ за умови використання відповідного програмного забезпечення.

В даний момент, для вирішення даного завдання бачиться можливим використання програми OMNet++, з подальшим розрахунком ефективності моделі.

2) розробка моделі бездротової сенсорної мережі.

3) налаштування моделі за допомогою обраних протоколів маршрутизації.

4) проведення експериментів для оцінки якості отриманої моделі та порівняння роботи протоколів.

Висновки до першого розділу

У першому розділі було проведено дослідження бездротових сенсорних мереж, їх зв’язку з IoT, методів їх моделювання та обчислювання їх ефективності. Було визначено необхідність попереднього створення моделей БСМ з метою їх перевірки та вдосконалення.

Виходячи з проаналізованих даних та стану галузі було сформульовано основні цілі магістерської роботи та обрано шлях їх досягнення.

Аналіз існуючого програмного забезпечення дав можливість обрання найбільш відповідного до цілей роботи програмного продукту (OMNet++). Використання даного ПЗ для моделювання БСМ має наступні переваги:

1. Можливість промоделювати будь-яку мережу, компоненти якої взаємодіють між собою за допомогою передачі повідомлень.
2. Підтримка основними платформами.
3. Велика кількість допоміжних матеріалів та відео уроків.
4. Підтримка моделювання великих мереж. Обмеження лише за можливостями комп’ютера, який використовується у ході моделювання.
5. Простий зрозумілий на інтуїтивному рівні інтерфейс.
6. Безкоштовне поширення ПЗ.
7. Повна реалізація основних функцій мережевого рівня ZigBee.

За допомогою обраного програмного забезпечення, математичних моделей та існуючих даних було вирішено створити модель функціональної БСМ, налаштувати з використанням протоколів маршрутизації, провести дослідження її відповідності до заданих вимог та порівняти функціональність моделі за умови використання різних протоколів. Разом із перевіркою функціональності та ефективності моделі буде проведено математичний аналіз обраного ПЗ на предмет доцільності його використання для вирішення подібних задач. За умови позитивних результатів експериментів створена модель може бути використана у реальних умовах.

Література до першого розділу

1. Беспроводные сенсорные сети – Режим доступу: www. URL: https://compress.ru/article.aspx?id=17950
2. O.V. Tuzhilkin, N.S. Ulianin METHODS OF EVALUATING THE PERFORMANCE OF WIRELESS SENSOR NETWORK
3. Налаштування мережі з використанням динамічного протоколу маршрутизації OSPF: Майбутній науковець – 2018: матеріали всеукр. наук.-практ. конф. з міжнар. участю (14 груд. 2018 р.), м. Сєвєродонецьк. Ч. II / укладач В. Ю. Тарасов – Сєвєродонецьк: Східноукр. нац. ун-т ім. В. Даля, 2018.
4. Analysis and modeling of dynamic routing network protocols: III International Conference TACSIT-2019 Proceedings, May 7-8, 2019, Severodonetsk, Ukraine / I. Skarga-Bandurova (Ed.). – Severodonetsk: Volodymyr Dahl East Ukrainian National University, 2019.
5. Wikipedia. IEE 802.15.4 – Режим доступу: www. URL: https://en.wikipedia.org/wiki/IEEE_802.15.4
6. IEEE 802.15 WPAN™ Task Group 4 (TG4) – Режим доступу: www. URL: http://www.ieee802.org/15/pub/TG4.html
7. Naagesh Bhat. IEEE 802.15.4 Protocol Implementation on FPGA // LAP Lambert Academic Publishing. – 2013.
8. БЕСПРОВОДНАЯ СЕНСОРНАЯ ПЛАТФОРМА ДЛЯ ТЕХНОЛОГИИ IoT ИНФОРМАЦИОННАЯ СРЕДА НА ГРАНИЦЕ СЕТИ – Режим доступу: www. URL: http://docplayer.ru/71006280-Besprovodnaya-sensornaya-platforma-dlya-tehnologii-iot-informacionnaya-sreda-na-granice-seti.html
9. Олифер В.Г. Олифер Н.А. «Компьютерные сети. Принципы, технологии, протоколы. Учебник для вузов. 4-е изд». С.-Пб.:"Питер", 2014.
10. Алгоритмы маршрутизации – Режим доступу: www. URL: http://citforum.ru/nets/ito/2.shtml.
11. Постольский С.П. ОБЗОР ПРОБЛЕМНЫХ ОБЛАСТЕЙ В БЕЗОПАСНОСТИ БЕСПРОВОДНЫХ СЕНСОРНЫХ СЕТЕЙ, АТАК И МЕХАНИЗМОВ ИХ ЗАЩИТЫ // Научное сообщество студентов XXI столетия. ТЕХНИЧЕСКИЕ НАУКИ: сб. ст. по мат. XXXII междунар. студ. науч.-практ. конф. № 5(31) – Режим доступу: www. URL: https://sibac.info/studconf/tech/xxxii/42203
12. СИСТЕМА МОДЕЛИРОВАНИЯ СЕТЕЙ СВЯЗИ NS-2 – Режим доступу: www. URL: https://docplayer.ru/33398038-Sistema-modelirovaniya-setey-svyazi-ns-2.html
13. Аппаратные и программные решения для беспроводных сенсорных сетей – Режим

доступу: www. URL: http://intuit.valrkl.ru/course-1240/

14. Скрипов С.А. Имитационное моделирование беспроводных сетей со сложной структурой // Молодой учёный. – 2010.
15. Скрипов С.А. Разработка протоколов маршрутизации для беспроводных сетей со специальной топологией // IV Международная научно-практическая конференция "Современные информационные технологии и ИТ образования". – 2010.
16. Ogier R.G. et al. Topology Broadcast based on Reverse-Path Forwarding (TBRPF) // INTERNET-DRAFT, MANET Working Group. – 2012.
17. Балонин Н. А., Сергеев М. Б. Беспроводные персональные сети на основе ZigBee. Учебное пособие. – СПб: ГУАП, 2015.

РОЗДІЛ 2
ДОСЛІДЖЕННЯ ВНУТРІШНЬОЇ СТРУКТУРИ, СТАНДАРТІВ ТА ПРОТОКОЛІВ БЕЗДРОТОВИХ СЕНСОРНИХ МЕРЕЖ

2.1. Структура та архітектура БСМ

Типова бездротова сенсорна мережа може бути розділена на два елементи: сенсорний вузол і мережева архітектура [18].

Сенсорний вузол бездротової сенсорної мережі

Сенсорний вузол в БСМ складається з чотирьох основних компонентів: джерело живлення, датчик, блок обробки, система зв'язку.

Рисунок 2.1 – Структура сенсорного вузла

Датчики (сенсори) вузлів мережі призначені для збору інформації про навколишнє (вузла) середовище. Датчики вузлів поділяють на [18]:

1. Пасивні: температури, інфрачервоні, вологості, акустичні, біохімічні, тиску та ін.;
2. Активні: ультразвукові, інфрачервоні з підсвічуванням, радіолокаційні та ін.

Датчик збирає аналогові дані з фізичного світу, АЦП перетворює ці дані в цифрові дані. Основний мікроконтролер виконує інтелектуальну обробку даних і маніпулювання ними.

Система зв'язку складається з системи радіозв'язку, зазвичай радіостанції ближньої дії, для передачі і прийому даних. Оскільки всі компоненти є пристроями з низьким енергоспоживанням, для живлення всієї системи використовується невелика батарея, така як CR-2032 [19].

Незважаючи на назву, сенсорний вузол виконує не тільки функції сенсорного компонента, а й з інші важливі функції, такі як обробку, зв'язок та зберігання. Завдяки всім цим функціям, компонентам і удосконаленням вузол датчика відповідає за збір даних фізичного світу, аналіз мережі, кореляцію даних і об'єднання даних другого датчика з власними даними.

Архітектура бездротової сенсорної мережі

Коли велика кількість сенсорних вузлів розгорнута у великій області для спільного моніторингу фізичного середовища, об'єднання в мережу цих сенсорних вузлів однаково важливо. Сенсорний вузол в БСМ не тільки зв'язується з іншими сенсорними вузлами, але також і з базовою станцією, використовуючи бездротовий зв'язок [19].

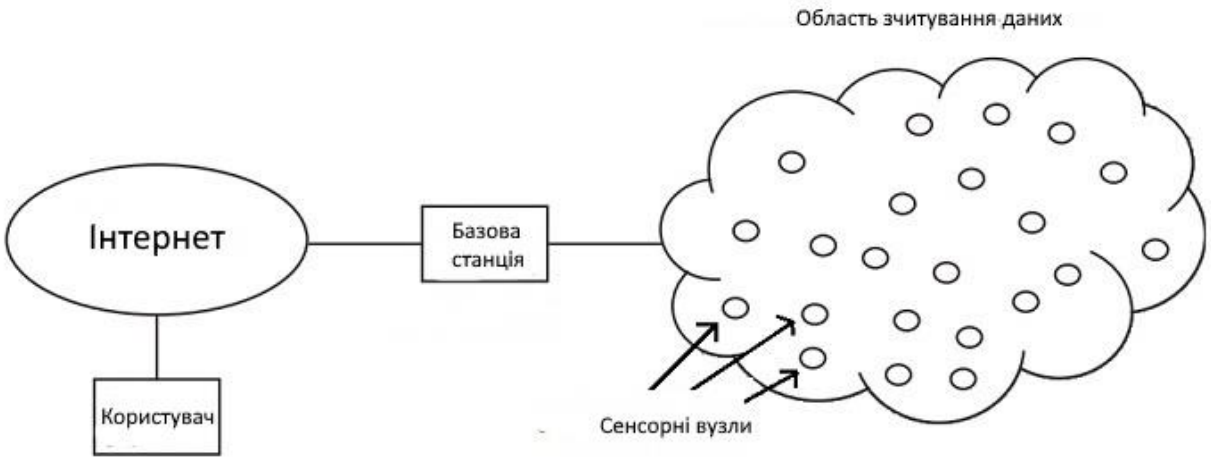


Рисунок 2.2 – Мережева архітектура

Базова станція відправляє команди на сенсорні вузли, а сенсорні вузли виконують завдання, взаємодіючи один з одним. Після збору необхідних даних сенсорні вузли відправляють дані в базову станцію.

Базова станція також діє як шлюз для інших мереж через Інтернет. Після прийому даних від вузлів датчиків базова станція виконує просту обробку даних і відправляє оновлену інформацію користувачеві через Інтернет [19].

Якщо кожен вузол датчика підключений до базової станції, то це архітектура мережі з одним переходом (або однострибкова архітектура). Хоча передача на великі відстані можлива, споживання енергії для зв'язку буде значно вищим, ніж для збору і обчислення даних.

Архітектури односкачкової та багатоскачкової мереж приведені на рис. 2.3 та 2.4.

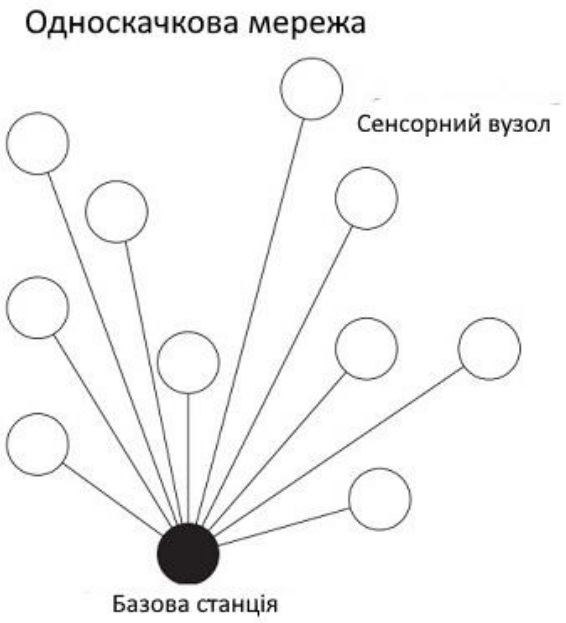


Рисунок 2.3 – Однострибкова мережа

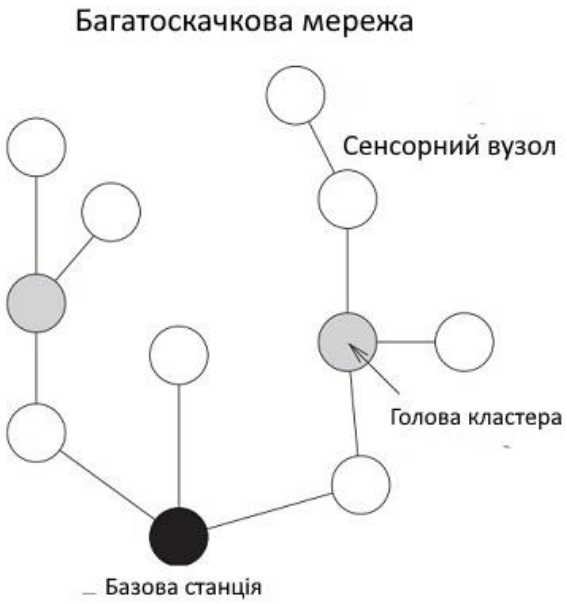


Рисунок 2.4 – Багатострибкова мережа

Отже, багатострибкова мережева архітектура зазвичай використовується в серйозних додатках. Замість однієї єдиної лінії зв'язку між вузлом датчика і базовою станцією дані передаються через один або кілька проміжних вузлів [19].

Це може бути реалізовано двома способами. Архітектура плоскої мережі та архітектура ієрархічної мережі. У плоскій архітектурі базова станція відправляє команди всім сенсорним вузлам, але сенсорний вузол з запитом, що збігається, відповість, використовуючи свої рівноправні вузли через багатострибковий шлях [20].

Рисунок 2.5 – Плоска архітектура

В ієрархічній архітектурі група сенсорних вузлів формується у вигляді кластера, і сенсорні вузли передають дані до відповідних голів кластера. Потім голови кластера можуть передавати дані на базову станцію [20].

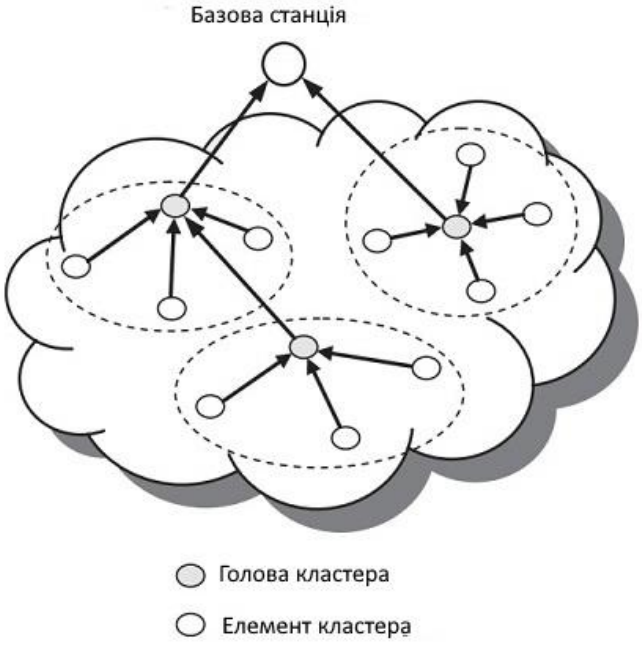


Рисунок 2.6 – Ієрархічна архітектура

2.2. Види вузлів

Типовий вузол може бути представлений трьома типами пристроїв [20]:

1. Мережевий координатор (Network Coordination Device);

- здійснює глобальну координацію, організацію та установку параметрів мережі;
- найбільш складний з трьох типів пристроїв, вимагає найбільшого обсягу пам'яті та джерело живлення.

2. Пристрій з повним набором функцій (Fully Function Device);
   - підтримка стандарту 802.15.4;
   - додаткова пам'ять і енергоспоживання дозволяє виконувати роль координатора мережі;
   - підтримка всіх типів топологій («точка-точка», «зірка», «дерево», «чарункова мережа»);
   - здатність виконувати роль координатора мережі;
   - здатність звертатися до інших пристроїв в мережі.
3. Reduced Function Device;
   - підтримує обмежений набір функцій 802.15.4;
   - підтримка топологій «точка-точка» та «зірка»;
   - не виконує функції координатора;
   - звертається до координатора мережі і маршрутизатора.

2.3. Класифікація БСМ

Бездротові сенсорні мережі надзвичайно специфічні для конкретного додатка і розгортаються відповідно до вимог програми. Отже, характеристики однієї БСМ будуть відрізнятися від характеристик іншої БСМ [20].

Незалежно від застосування, бездротові сенсорні мережі в цілому можна класифікувати за наступними категоріями.

- Статична і мобільна БСМ.
- Детермінована і недетермінірована БСМ.
- З однієї базовою станцією і декількома базовими станціями.
- Зі статичними базовими станціями і мобільними базовими станціями.
- Однострибкова і багатострибкова БСМ.
- Самоналагоджувальна і неконфігуріруєма БСМ.
- Гомогенна і гетерогенна БСМ.

2.4. Топології БСМ

БСМ може бути мережею з одним або декількома вузлами. Нижче наведено кілька різних мережевих топологій, які використовуються в БСМ.

У топології «зірка» існує один центральний вузол, відомий як концентратор або комутатор, і кожен вузол в мережі підключений до цього концентратора. Топологія «зірка» дуже проста в реалізації, проектуванні і розширенні. Оскільки всі дані проходять через концентратор, вони грають важливу роль в мережі, і збій в концентраторі може привести до відмови всієї мережі [20].

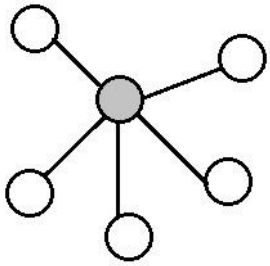


Рисунок 2.7 – Топологія «зірка»

Топологія «дерево» – це ієрархічна мережа, в якій зверху знаходиться один кореневий вузол, і цей вузол підключений до багатьох вузлів на наступному рівні, і це триває далі. Потужність обробки і енергоспоживання є найвищими в кореневому вузлі і продовжують зменшуватися в міру зниження ієрархічного порядку.

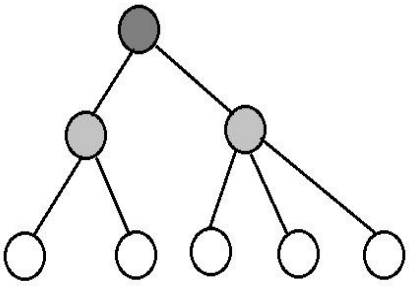


Рисунок 2.8 – Топологія «дерево»

У mesh-топології (топологія сітки), крім передачі своїх власних даних, кожен вузол також діє як ретранслятор для передачі даних інших підключених вузлів. Топології сітки далі діляться на повністю підключену сітку і частково підключену сітку. У повністю пов'язаній топології сітки кожен вузол пов'язаний з кожним іншим вузлом, в той час як в частково пов'язаної топології сітки вузол пов'язаний з одним або декількома сусідніми вузлами [19].

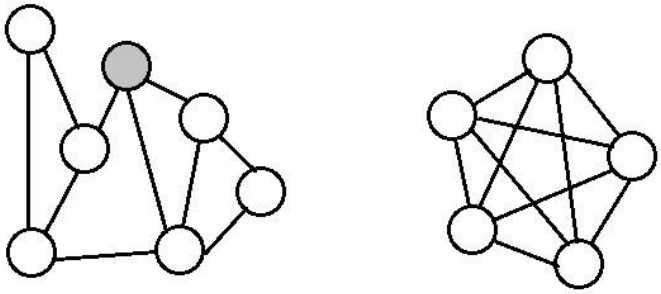


Рисунок 2.9 – Топології «сітка»

2.5. Стандарти на основі IEEE 802.15.4

2.5.1. Стандарт ZigBee

ZigBee – це відкритий стандарт, що використовується у БСМ. Технологія ZigBee надає можливість створювати самоорганізовані і самовідтворювані бездротові мережі з ретрансляцією повідомлень, що проводиться автоматично, а також підтримкою мобільних та батарейних вузлів [21].

В даний час технологія ZigBee виходить за межі дослідницьких лабораторій і починає ефективно застосовуватися для створення бездротових мереж датчиків, охоронних систем, систем автоматизації будівель та інших систем керування.

Стандарт ZigBee передбачає частотні канали в діапазонах 868 МГц, 915 МГц і 2,4 ГГц. Найбільші швидкості передачі даних і найвища стійкість досягаються в діапазоні 2,4 ГГц [21]. Тому більшість виробників мікросхем випускають приймачі саме для цього діапазону, в якому передбачено 16 частотних каналів з кроком 5 МГц.

Швидкість передачі даних разом зі службовою інформацією в ефірі становить близько 250 кбіт/с [21]. При цьому середня пропускна спроможність вузла для корисних даних, в залежності від завантаженості мережі та кількості ретрансляцій, може лежати в межах 5 ... 40 кбіт/с [21].

Відстані між вузлами мережі складають десятки метрів під час роботи всередині приміщення і сотні метрів на відкритому просторі. За рахунок ретрансляції зона покриття мережі може значно збільшуватися [22].

В основі мережі ZigBee лежать Mesh-мережі (mesh-топологія). У такій мережі, кожен пристрій може зв'язуватися з будь-яким іншим пристроєм як безпосередньо, так і через проміжні вузли мережі. Mesh-мережі пропонують альтернативні варіанти вибору маршруту між вузлами. Повідомлення передаються між вузлами, до того моменту, поки не будуть доставлені кінцевому одержувачу. Шляхи проходження повідомлень можуть сильно різнитися, що

підвищує доступність мережі в разі виходу з ладу тієї чи іншої ланки.

Альянс ZigBee, заснований в 2002 році, являє собою співтовариство компаній (вже більше 300), які об'єдналися з метою розробки ефективних протоколів для бездротової мережі і забезпечення взаємодії між різними пристроями різних виробників.

Тепер зупинимося на структурі самої мережі ZigBee і типах пристроїв, які в ній можуть бути [23].

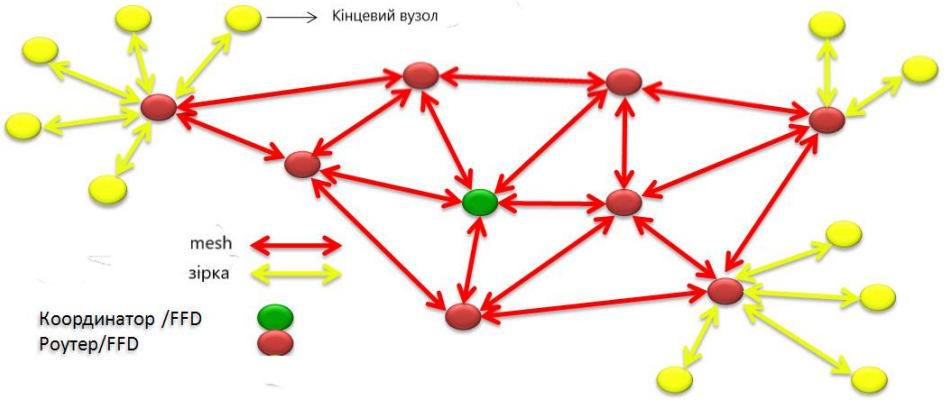


Рисунок 2.10 – Типова структура мережі ZigBee

Координатор – це вузол, який організував мережу. Саме він вибирає політику безпеки мережі, дозволяє або забороняє підключення до мережі нових пристроїв, а також при наявності перешкод в радіоефірі ініціює процес переводу всіх пристроїв в мережі на інший частотний канал.

Роутер – це вузол, який має стаціонарне живленняі отже може постійно брати участь в роботі мережі. Координатор також є роутером. На вузлах цього типу лежить відповідальність за маршрутизації мережевого трафіку. Роутери постійно підтримують спеціальні таблиці маршрутизації, які використовуються для прокладки оптимального маршруту і пошуку нового, якщо раптом якийсь пристрій вийшов з ладу. Наприклад, роутерами в мережі ZigBee можуть бути «розумні» розетки, блоки керування освітлювальними приладами або будь-який інший пристрій, який має підключення до мережі електроживлення [23].

Кінцевий пристрій – це пристрій, який підключається до мережі через батьківський вузол – роутер або координатор – і не бере участі в маршрутизації трафіку. Все спілкування з мережею для них обмежується передачею пакетів на «батьківський» вузол або зчитуванням даних, що надійшли з нього ж. «Батьком» для таких пристроїв може бути будь-який роутер або координатор. Кінцеві пристрої більшу частину часу перебувають у сплячому режимі і відправляють керуюче або інформаційне повідомлення зазвичай тільки по певній події (натискання кнопки вимикача, відкриття вікна або двері). Це дозволяє їм довго зберігати

енергію вбудованого джерела живлення [23]. Прикладом кінцевих пристроїв в мережах ZigBee можуть бути бездротові вимикачі, які керують роботою світильників і працюють від батарейок, датчики протікання води, датчики відкриття/закриття дверей. Варто сказати, що кінцеві пристрої діляться на 3 категорії, кожна з яких має свої особливості.

2.5.2. Стандарт WirelessHART або IEC 62591

WirelessHART або IEC 62591 – мережева технологія для бездротових пристроїв на базі протоколу HART (Highway Addressable Remote Transducer Protocol).

Протокол використовує синхронізовану в часі, що самоорганізується і самовідновлюється, коміркову архітектуру. Він працює в діапазоні частот 2400-2483,5 МГц для промислової, медичної та наукової апаратури (ISM) стандарту IEEE 802.15.4. Устаткування, побудоване на даному стандарті відноситься до пристроїв малого радіусу дії. Розроблений як інтероперабельний бездротовий стандарт, що підтримує обладнання різних виробників, WirelessHART був створений відповідно до вимог мереж польових пристроїв. В основу протоколу лягла технологія TSMP (синхронізований у часі пористий протокол) компанії Dust Networks [24].

Влітку 2009 року NAMUR, міжнародна асоціація користувачів в хімічній і фармацевтичній галузях, провела польові випробування WirelessHART на предмет відповідності вимогам NAMUR щодо бездротових засобів автоматизації.

У квітні 2010 року протокол WirelessHART був одноголосно затверджений представниками Міжнародної електротехнічної комісії (МЕК) і таким чином став першим бездротовим міжнародним стандартом, під номером IEC 62591 [24].

2.5.3. Стандарт ISA100.11a

SA100.11a – стандарт технології бездротових мереж, розроблений Міжнародним товариством автоматизації (ISA). Офіційний опис – "Бездротові системи для промислової автоматизації: керування процесами та супутними програмами" [25].

У 2009 році Інститут відповідності стандартам автоматизації ISA створив Інститут бездротової відповідності ISA100. Інститут бездротової відповідності ISA100 володіє схемою сертифікації «Відповідність ISA100», яка забезпечує незалежне тестування продуктів на базі ISA100, щоб переконатися, що вони відповідають стандарту ISA100 [25].

2.5.4. Стандарт MiWi

MiWi – фірмовий бездротовий протокол, що підтримує зв'язок Peer-to-Peer Star і Mesh, розроблений Microchip Technology, використовує невеликі цифрові радіостанції малої потужності на основі стандарту IEEE 802.15.4. Він призначений для мереж з низькою потужністю, з обмеженими витратами, таких як промисловий моніторинг та керування, автоматизація будинків і будинків, дистанційне керування, бездротові датчики, керування освітленням та автоматичне зчитування лічильника [26].

Протокол MiWi підтримується на пристроях і модулях на базі SAMR30 (Sub-GHz) та SAMR21 (2,4 ГГц) ARM Cortex-M0 + Microchip. Код протоколу MiWi, що підтримується мікроконтролерами PIC та dsPIC, був заморожений і більше не рекомендується для нових конструкцій; однак вона все ще доступна в Бібліотеці мікрочіпів для додатків (MLA) для інтегрованого середовища розробки MPLAB.

Найновіший підтримуваний стек протоколів доступний безкоштовно в розширеному програмному забезпеченні для інтегрованого середовища розвитку Atmel Studio.

Microchip випустив Посібник із швидкого старту, Посібник з проектування програмного забезпечення MiWi та Посібник з міграції, в якому представлена технічна інформація про MiWi. Це не в першу чергу специфікації протоколів і зосереджені на впровадженні протоколу MiWi на мікроконтролерах Microchip [26].

Протокол MiWi - це невелика альтернатива ZigBee (40K-180K), яка корисна для економічних програм із обмеженою пам’яттю [26]. Хоча програмне забезпечення MiWi можна безкоштовно завантажити з його офіційного веб-сайту, це рішення, яке потребує використання лише з мікроконтролерами Microchip.

Стек протоколу MiWi підтримує бездротову мережу топологій зірок та однорангових мереж, корисних для простого, короткого, бездротового зв'язку між вузлом та вузлом. Крім того, стек забезпечує функції сну, активного сканування та виявлення енергії, підтримуючи вимоги до низької потужності пристроїв, що працюють на батареях.

2.6. Протоколи маршрутизації для БСМ

2.6.1. Основні протоколи БСМ

Протоколи маршрутизації в БСМ вирішують наступні завдання:

- Самоорганізація вузлів мережі (самоконфігурірування, самовідновлення та оптимізація).
- Маршрутизація пакетів даних і адресація вузлів.
- Мінімізація енергоспоживання вузлів мережі і збільшення загального часу життя всієї

мережі.

- Збір і агрегація даних.
- Регулювання швидкості передачі і обробки даних в мережі.
- Максимізація зони покриття мережі.
- Якість обслуговування (QoS).
- Захист від несанкціонованого доступу.

Протоколи маршрутизації БСМ відповідають за підтримку маршрутів в мережі і повинні гарантувати надійний зв'язок навіть в жорстких несприятливих умовах. Багато протоколів маршрутизації були спеціально розроблені для БСМ, де енергозбереження є суттєвою проблемою, на вирішення якої спрямований протокол. Інші ж були розроблені для загального застосування в бездротових мережах, але знайшли своє застосування і в БСМ. У табл. 2.1 наведено основні протоколи БСМ за категоріями:

Таблиця 2.1 – Класифікація протоколів маршрутизації БСМ

Категорія протоколів	Протоколи
Засновані на місцезнаходженні вузлів	SMECN, GEAR, Span, TBF, BVGF, GeRaF, MECN, GAF, Span, BVGF
Направлені на агрегацію даних	Directed Diffusion, COUGAR, ACQUIRE, EAD, SPIN, Rumor Routing, EAD
Ієрархічні	LEACH, TEEN, APTEEN, PEGASIS, HEED
Засновані на мобільності	Joint Mobility and Routing, TTDD, Data Mules, Dynamic Proxy Tree-Base Data Dissemination, SEAD
Мульти-орієнтовані	Sensor-Disjoint Multipath, N-to-1 Multipath Discovery, Braided Multipath
Засновані на гетерогенності	IDSQ, CHR, CADR
Засновані на якості обслуговування	SAR, Energy-aware routing, SPEED

У даній роботі будуть розглядатися та порівнюватися наступні протоколи: LEACH, TEEN, HEED, які належать до категорії кластер них протоклів, що будуть розглядатися далі.

2.6.1.1. Кластерні протоколи

Багатьма дослідників було доведено, що протоколи маршрутизації на основі кластерів є більш енергоефективними, ніж протоколи маршрутизації, що не базуються на кластерах, тим, що вони збільшують масштабованість та тривалість життя мережі [28].

Протоколи маршрутизації на основі кластерів складаються з раундів. Кожен раунд складається з чотирьох етапів: вибір голови кластера, формування кластера, внутрішньокластерна комунікація та комунікація між кластерами. На першому етапі

обираються вузли голів кластера (CH) з метою збору даних від членів кластера (CM) або звичайних вузлів. Після цього етапу слід побудувати кластери. Іншими словами, звичайні вузли вибирають CH для приєднання та формування кластерів. На етапі комунікації між кластерами CM збирають дані та надсилають їх до своїх CH. Після отримання даних від CM, CH виконують агрегацію даних, щоб опустити зайві дані. На етапі комунікації між кластерами CH надсилають дані до БС Фаза комунікації як внутрішньої, так і між кластерної може бути виконана в режимі одночасного або багатоспільного зв'язку.

Протоколи кластеризації мають численні переваги, які роблять їх найбільш підходящими та сумісними протоколами для WSN [28]:

- Загальні витрати енергії на передачу даних мінімізовані.
- Зменшується зіткнення даних та усуваються зайві дані в процесі агрегації даних.
- Затримка зменшується, оскільки завдання передачі даних виконують лише CH.
- Попит на пропускну здатність зменшується, а обмежена пропускна здатність ефективно використовується.
- Зменшуються витрати на маршрутизацію та утримання топології.
- Покращується керованість мережі та масштабованість

У свою чергу кластерні протоколи розподіляються на окремі типи, що представлені на рис. 2.11 [29].

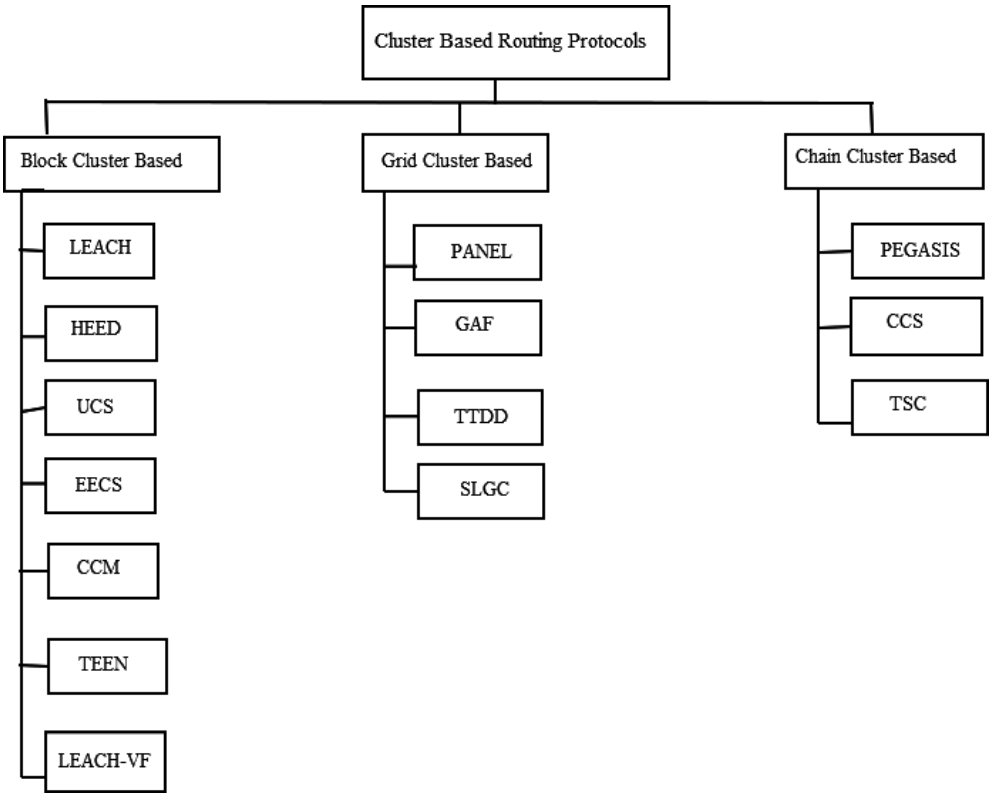


Рисунок 2.11 – Типи кластерних протоколів

2.6.1.1.1. Протокол LEACH

LEACH – це самоорганізуючий, адаптивний протокол кластеризації, який використовує рандомізацію для рівномірного розподілу енергетичного навантаження між датчиками в мережі та використовується, як простий протокол маршрутизації в бездротових сенсорних мережах (БСМ). Мета LEACH – знизити споживання енергії, необхідну для створення та підтримки кластерів, щоб покращити час роботи бездротової сенсорної мережі [30].

LEACH – це ієрархічний протокол, в якому більшість вузлів передають головам кластерів, а голови кластерів агрегують і стискають дані та передають їх на базову станцію. Кожен вузол використовує стохастичний алгоритм для кожного раунду, щоб визначити, чи стане він головою кластера в цьому раунді [31]. LEACH передбачає, що кожен вузол має радіостанцію, достатньо потужну, щоб безпосередньо дістатися до базової станції або найближчої голови кластера, але використання цієї радіостанції на повну потужність весь час буде витрачати енергію.

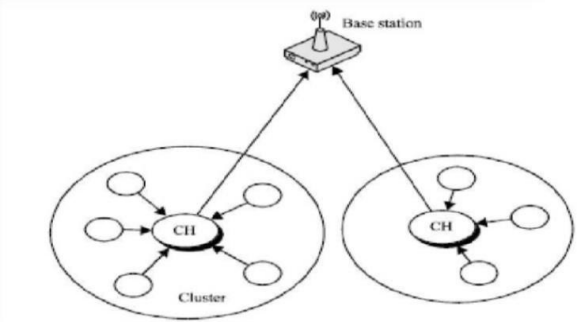


Рисунок 2.12 – Структура LEACH

У LEACH вузли організовуються в локальні кластери, причому один вузол діє як локальна базова станція або заголовок кластера. LEACH включає рандомізоване обертання положення високої енергії кластера-голівки таким чином, що воно обертається між різними датчиками, щоб не розрядити акумулятор одного датчика. Крім того, LEACH здійснює локальний синтез даних для «стиснення» кількості даних, що надсилаються з кластерів на базову станцію, ще більше скорочуючи розсіювання енергії та збільшуючи термін служби системи [32].

Датчики обирають себе локальними головами кластерів у будь-який момент з певною вірогідністю. Кожен вузол приймає рішення про те, чи бути головою кластера незалежно від інших вузлів у мережі, і тому додаткові переговори не потрібні для визначення кластерів-головок. Ці вузли кластера передають свій статус іншим датчикам у мережі. Кожен вузол датчика визначає, до якого кластеру він хоче належати, вибираючи голову кластера, яка

потребує мінімальної енергії зв'язку. Після того як всі вузли організовані в кластери, кожна голова кластера створює графік для вузлів у своєму кластері. Це дозволяє радіокомпонентам кожного некластерного вузла вимикатись у будь-який час, за винятком його часу передачі, мінімізуючи енергію, що розсіюється в окремих датчиках. Після того, як заголовок кластера має всі дані з вузлів свого кластера, вузол голови кластера агрегує дані і потім передає стислі дані на базову станцію [33].

Апріорі система може визначати оптимальну кількість кластерів, які мають мати в системі. Це залежатиме від кількох параметрів, таких як топологія мережі та відносні витрати на обчислення та комунікацію. Якщо менше, ніж оптимальної кількості кластерних голів, деяким вузлам в мережі доведеться передавати свої дані дуже далеко, щоб досягти голови кластера, внаслідок чого глобальна енергія в системі буде великою. Якщо існує більше оптимальної кількості головок кластерів, то відстані вузлів, які повинні пройти, щоб досягти найближчої голови кластера, не зменшуються істотно, але є більше головок кластерів, які повинні передавати дані на великі відстані до базової станції, і місцеве стиснення проводиться менше [33]. Окрім зменшення розсіювання енергії, LEACH успішно розподіляє енергоспоживання між вузлами мережі таким чином, що вузли гинуть випадковим чином і з однаковою швидкістю.

Усі вузли, які не є головами кластерів, спілкуються з головою кластера лише відповідно до розкладу, створеного головою кластера. Вони роблять це, використовуючи мінімальну енергію, необхідну для досягнення голови кластера, і потрібно лише утримувати радіоприймачі протягом свого проміжку часу.

Властивості цього алгоритму включають:

- Вибір голови кластера кожен раунд з обертанням. Або вибір голови кластера на основі датчика, що має найбільшу енергію.
- Адаптивне членство в кластері.
- Збір даних на голові кластера.
- Голова кластера спілкується безпосередньо з раковиною або користувачем.
- Зв'язок здійснюється з головою кластера через TDMA.

TDMA (англ. Time Division Multiple Access – множинний доступ з поділом за часом) – спосіб використання радіочастот, коли в одному частотному інтервалі знаходяться кілька абонентів, різні абоненти використовують різні тимчасові слоти (інтервали) для передачі. Є додатком мультиплексування каналу з поділом за часом (TDM - Time Division Multiplexing) до радіозв'язку.

До недоліків LEACH належать [34]:

- Залишок енергії серед вузлів не враховується при виборі голови кластерів.
- Довільні та змінні розміри кластерних утворень.
- Випадковий і нерівномірний розподіл головок кластера.
- Комунікація з одноразовим переходом у ситуаціях, коли використання енергії менш ефективно від голови кластера до базової станції.

Операція LEACH розбита на раунди, де кожен раунд починається з фази налаштування, коли кластери організовані, а потім фаза стаціонарного стану, коли відбувається передача даних на базову станцію. Щоб мінімізувати накладні витрати, стаціонарна фаза довга порівняно з фазою налаштування.

1. Фаза реклами

Спочатку, коли створюються кластери, кожен вузол вирішує, чи слід стати головою кластера для поточного раунду. Це рішення засноване на пропонованому відсотку головок кластерів для мережі (визначається апріорі) та кількості разів, коли вузол був головою кластера до цього часу. Це рішення приймається вузлом n, вибираючи випадкове число між 0 і 1.

Якщо число менше порогу T (n), вузол стає головою кластера для поточного раунду. Поріг встановлюється як [35]:

$$T(n) = \frac{p}{1 - p * (r \bmod \frac{1}{p})} : if n \in G$$

$$T(n) = 0 \text{ в інших випадках} \qquad (2.1)$$

де P – бажаний відсоток голів кластерів, r – поточний раунд, а G – сукупність вузлів, які не були головами кластерів в останніх 1/P раундах. Використовуючи цей поріг, кожен вузол буде головою кластера в певний момент в межах 1/P раундів. Вузли, які є головами кластерів у раунді 0, не можуть бути головами кластерів для наступних 1/P раундів. Таким чином, ймовірність того, що решта вузлів є головами кластерів, повинна бути збільшена, оскільки існує менше вузлів, які можуть стати головами кластерів. Кожен вузол, який обрав себе головою кластера для поточного раунду, передає рекламне повідомлення решті вузлів. Для цього етапу «кластерна голова-реклама», голови кластерів використовують протокол MAC CSMA, і всі голови кластерів передають свою рекламу, використовуючи однакову енергію передачі. Вузли без кластерної голови повинні підтримувати приймачі під час цієї фази налаштування, щоб слухати рекламу всіх вузлів голови кластера. Після завершення цієї фази кожен вузол, який не є головним кластером, визначає кластер, якому він буде належати для цього раунду.

Це рішення базується на силі прийнятого сигналу реклами. У випадку зв’язків вибирається випадкова голова кластера [34].

2. Фаза налаштування кластера

Після того, як кожен вузол вирішить, до якого кластеру він належить, він повинен повідомити вузол голови кластера, що він буде членом кластера. Кожен вузол знову передає цю інформацію в голову кластера за допомогою протоколу MAC CSMA. Під час цієї фази всі вузли голови кластера повинні утримувати свої приймачі.

3. Створення розкладу

Кластерний вузол приймає всі повідомлення для вузлів, які хотіли б включити до кластеру. На основі кількості вузлів кластера, вузол голови кластера створює графік TDMA, який повідомляє кожному вузлу, коли він може передавати. Цей графік передається назад до вузлів кластера.

4. Передача даних

Після створення кластерів і виправлення розкладу TDMA може розпочатись передача даних. Припускаючи, що вузли завжди мають дані для надсилання, вони надсилають їх протягом відведеного часу передачі до голови кластера. Ця передача використовує мінімальну кількість енергії (вибирається виходячи з прийнятої сили реклами кластерної голови). Радіо кожного вузла без кластерної голови може бути вимкнено до виділення вузлом часу передачі, таким чином мінімізуючи розсіювання енергії в цих вузлах. Вузол голови кластера повинен підтримувати його приймач, щоб отримувати всі дані з вузлів кластера. Після отримання всіх даних вузол голови кластера виконує функції обробки сигналу для стиснення даних в єдиний сигнал [35]. Цей складений сигнал відправляється на базову станцію. Через певний час, який визначається апріорі, наступний раунд починається з кожного вузла, визначаючи, чи повинен він бути головою кластера для цього раунду та рекламувати цю інформацію, як описано в пункті 1.

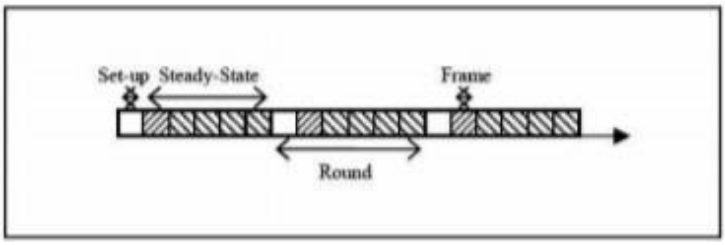


Рисунок 2.13 – Фази протоколу LEACH

2.6.1.1.2. Протокол TEEN

TEEN – протокол маршрутизації реактивної кластеризації. Голова кластера (CH) кожного кластера збирає дані від своїх членів кластера. Голови кластера обробляють дані та передають дані в базові станції або голови кластерів вищого рівня. Усі вузли потребують

передачі даних лише до своєї голови кластера, і лише голови кластерів повинні агрегувати дані, що є перевагою кластеризованих протоколів маршрутизації. Це сприяє енергозбереженню. Всі вузли по черзі стають головами кластера, щоб рівномірно розподілити споживання енергії. Для виборів голови кластеру використовується механізм випадкового відбору. Після формування кластерів голови кластерів виділяють часовий проміжок для членів кластера, в якому члени кластера можуть передавати свої дані [36].

У протоколі TEEN процес формування голови кластера базується на LEACH. Спочатку формуються кластери, а потім голова кластера транслює два пороги до всіх вузлів-членів: жорсткий поріг (HT) і м'який поріг (ST). У кожний час зміни кластера голова кластера також транслює ці два атрибути.

Функціонування TEEN:

- Жорсткий поріг (HT): це порогове значення для атрибута почуття. Це абсолютне значення атрибута, поза яким, вузол, який відчуває це значення, повинен увімкнути передавач і повідомити його голові кластера.
- М'який поріг (ST): Це невелика зміна значення атрибута відчуття, що запускає вузол, щоб він увімкнув передавач і передав його голові кластера.

Як йдеться у визначенні, лише тоді, коли атрибут почуття знаходиться в діапазоні, що цікавить, жорсткий поріг дозволяє вузлам передавати дані і тим самим вони значно зменшують кількість передач [37].

М'який поріг також значно зменшує кількість передачі почутих даних, оскільки він виключає передачу даних, якщо в атрибуті відчуття є незначні зміни або відсутні.

У цьому підході на основі інтересу кінцевого користувача вузли датчиків передаватимуть дані лише на основі жорсткого порогового значення та м'якого порогового значення, що призводить до більшої економії енергії. Ці два значення атрибута можуть бути допоміжними.

2.6.1.1.3. Протокол HEED

Hybrid Energy Efficient Distributed clustering (HEED) – протокол, який використовується для кластеризації і приносить енергоефективну маршрутизацію кластеризації з чітким врахуванням енергії. HEED не обирає головку кластера випадковим чином. HEED був розроблений для вибору різних головок кластера в полі відповідно до кількості енергії, яка розподіляється відносно сусіднього вузла. При цьому вибір головки кластера в першу чергу ґрунтується на залишковій енергії кожного вузла [38]. У цій залишковій енергії можна оцінити, знаючи енергію, споживану за біт для зондування, обробки та зв'язку. При цьому для розриву

зв'язків внутрішньокластерна комунікаційна вартість вважається вторинним параметром, а прив'язка означає, що вузол потрапляє в діапазон більш ніж однієї голови кластера. У цьому внутрішньокластерному зв'язку вартість розглядається як вторинний параметр. Процес кластеризації розділений на ряд ітерацій. Він заснований на двох параметрах [39].

- Перший параметр залежить від залишкової енергії вузла.
- Другий параметр - це врахування вартості зв'язку з кластерами.

Фази HEED:

- Фаза ініціалізації.
- Фаза обробки.
- Фаза завершення.

На етапі ініціалізації кожен вузол датчика встановлює значення ймовірності стати головою кластера (CHprob) на основі залишкової та максимальної енергії [40].

$$CH_{prob} = c_{prob} * \frac{E_{res}}{E_{\max}} \tag{2.2}$$

Де:

C_{prob} – початкова частка CH серед усіх датчиків.

E_{res} – поточна енергія в датчику.

E_{max} – максимальна енергія.

На етапі повторення вузол датчика знаходить голову кластера, до якого він може передавати дані з мінімальною енергією. Припустимо, якщо вузол датчика не отримав жодного повідомлення від голови кластера, він обирає себе як одну із голів кластера. Це повідомлення. Один - орієнтовний стан, коли вузол датчика стає орієнтовною головою кластера з CH_{prob} менше 1. А вузол датчика має остаточний статус, тобто він стає постійною головою кластера, коли його CH_{prob} досягає 1. На етапі завершення, вузол датчика визначає голову кластера, до якої він повинен підключитися [41]. Вузол датчика передає свою інформацію в голову кластера, і голова кластера відправляє агреговані дані в базову станцію в мультишопному режимі. HEED має такі переваги: Енергія вважається головним критерієм для того, щоб сенсорний вузол був обраний як голова кластера. Між кластерним зв’язком дозволяється тим самим зменшувати передачу на великі відстані, що значно економить енергію. Забезпечується рівномірний розподіл голів кластера по всій мережі, що забезпечує збалансування навантаження. Деякі недоліки HEED полягають у наступному: іноді ітерації виконуються для формування кластерів, що призводить до значних накладних витрат, і це споживає велику кількість енергії. Деякі CH, які знаходяться біля границь, швидко вичерпуються, і проблема перенавантаження зберігається [42].

2.7. Порівняння протоколів за основними ознаками

У табл. 2.2 наведено вибіркову порівняльну характеристику обраних для моделювання протоколів.

Таблиця 2.2 – Порівняльна таблиця протоколів

Ознака	**LEACH**	**TEEN**	**HEED**
Енергоефективність	Дуже низька	Дуже висока	Середня
Затримка	Дуже невелика	Невелика	Середня
Стабільність кластера	Середня	Висока	Висока
Масштабованість	Дуже низька	Низька	Середня
Врівноваження навантаження	Середня	Висока	Середня
Складність алгоритму	Низька	Висока	Середня

Висновки до другого розділу

У другому розділі проведено детальний аналіз структури та принципів роботи бездротових сенсорних мереж. Внутрішня структура проаналізована та проілюстрована у схематичному вигляді. Було розглянуто основні типи топологій, які використовуються для побудови БСМ.

Докладно проаналізована інформація щодо стандартів, як основи програмування БСМ.

Найважливішим питанням, яке вирішено у даному розділі – обрання протоколів маршрутизації для моделювання БСМ. Було обрано тип протоколів, які будуть використовуватися у процесі моделювання – кластерні протоколи. Такими протоколами обрані:

- LEACH;
- TEEN;
- HEED.

Ці протоколи були обрані, як одні з найвикористовуваніших. Було проаналізовано їх принципи роботи та складено порівняльну таблицю трьох протоколів. Крім того, їх базованість на кластерному підході дозволяє створити моделі мережі з максимально спільними умовами та характеристиками.

Результати, які були отримані при написанні другого розділу, були використані для створення, налаштування та порівняння трьох моделей БСМ на базі раніше обраного середовища моделювання – OMNeT++.

Література до другого розділу

18. ДЯДЮНОВ А.Н., КУЗНЕЦОВ К.Н. МОДЕЛИРОВАНИЕ БЕСПРОВОДНЫХ СЕНСОРНЫХ СЕТЕЙ // НАУЧНЫЙ ВЕСТНИК МГТУ ГА № 139. – 2010.
19. Что такое беспроводная сенсорная сеть (WSN): структура, классификация, топологии – Режим доступу: www. URL: http://digitrode.ru/articles/2015-chto-takoe-besprovodnaya-sensornaya-set-wsn-struktura-klassifikaciya-topologii.html
20. Современные и перспективные технологии построения беспроводных сенсорных сетей – Режим доступу: www. URL: https://lektsii.org/16-1692.html
21. Беспроводные сети ZigBee и Thread – Режим доступу: www. URL: http://www.wless.ru/technology/?tech=1
22. Основні поняття про ZigBee – Режим доступу: www. URL: https://crossgroup.su/solutions/data_transfer/zigbee.html
23. Варгаузин В.А. Сетевая технология ZigBee // ТелеМультиМедиа. 2015. № 6
24. Вікіпедія. WirelessHART – Режим доступу: www. URL: https://ru.wikipedia.org/wiki/WirelessHART
25. Wikipedia. ISA100.11a – Режим доступу: www. URL: https://en.wikipedia.org/wiki/ISA100.11a
26. ISA100.11a – Режим доступу: www. URL: https://isa100wci.org/
27. Wikipedia. MiWi – Режим доступу: www. URL: https://en.wikipedia.org/wiki/MiWi
28. A Survey on Cluster-based Routing Protocols in Wireless Sensor Networks – Режим доступу: www. URL: https://scialert.net/fulltextmobile/?doi=jas.2014.2011.2022
29. An Energy Centric Cluster-Based Routing Protocol for Wireless Sensor Networks – Режим доступу: www. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5982798/
30. Girish Prabhu, Dr.G.R.Udupi. Enhanced Leach Protocol // ISSN (e): 2250 – 3005 || Volume, 05 || Issue, 05 || May – 2015 || International Journal of Computational Engineering Research (IJCER)
31. Dr. R. R. Mudholkar, Dr. S. R. Sawant, V. C. Patil, Rjshree. V. Biradar “Classification & Comparison of Routing Protocols in WSN” UbiCC Journal- Vol 4
32. Sunil Pariyani, Rajesh Patel, Vijay Ukani, «Energy and Throughput Analysis of Hierarchical Routing Protocol(LEACH) for Wireless Sensor Networks», International Journal of Computer Applications Volume 20- No. 4 (April 2011)
33. Geetu, Sonia Juneja. Performance Analysis of SPIN and LEACH Routing Protocol in WSN // International Journal Of Computational Engineering Research (ijceronline.com) Vol. 2 Issue. 5

34. V.Vasanthi, «A Perspective analysis of routing protocol in wireless sensor network» (IJCSE) International Journal on Computer Science and Engineering Vol. 02.No. 08,2010,2511-2518.
35. K. Padmanabhan, «A Study on Energy Efficient Routing Protocols in Wireless Sensor Networks» European Journal of Scientific Research ISSN 1450-216X Vol.60 No.4 (2011), pp. 499-511 © EuroJournals Publishing, Inc. 2011.
36. SWATI R. MARKAD, DEVEN PATIL, SHAHRUKH, MOHD AYAZ KHAN. Data Centric Directed Diffusion Based On Weighted Grover's Quantum Algorithm In Wireless Sensor Network // International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 5 Issue 3, March 2016
37. Deepesh Dewangan, Srikant Singh. Analysis of Flooding and Directed Diffusion Protocol // International Journal of Science and Research (IJSR) ISSN (Online): 2319-7064 Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438
38. Shuaib, A.H.; Mahmoodi, T.; Aghvami, A.H.; A timed Petri Net model for the IEEE 802.15.4 CSMA-CA process, Personal, Indoor and Mobile Radio Communications, 2009 IEEE 20th International Symposium , Page(s): 1204 – 1210, 2009
39. S. Tilak et al., «A Taxonomy of Wireless Microsensor Network Models» in ACM Mobile Computing and Communications Review (MC2R), June 2012.
40. Ossama Younis and Sonia Fahmy «HEED: A Hybrid, Energy-Efficient, Distributed Clustering Approach for Ad-hoc Sensor Networks», Department of Computer Sciences, Purdue University 250 N. University Street, West Lafayette, IN 47907–2066, USA
41. Sasikumar M, Dr. R. Anitha «Performance Evaluation of Heterogeneous - HEED Protocol for Wireless Sensor Networks», International Journal of Advanced Research in Computer and Communication Engineering Vol. 3, Issue 2, February 2014
42. Bhawna Sharma, Anurag Rana «Energy Efficiency in Distributed Clustering of WSN using HEED Protocol», International Journal of Engineering Research & Technology (IJERT) IJERTV5IS060735 (This work is licensed under a Creative Commons Attribution 4.0 International License.) Vol. 5 Issue 06, June 2016

РОЗДІЛ 3
МОДЕЛЮВАННЯ БСМ ТА НАЛАШТУВАННЯ МОДЕЛЕЙ З ВИКОРИСТАННЯМ ПРОТОКОЛІВ МАРШРУТИЗАЦІЇ. АНАЛІЗ ОТРИМАНИХ РЕЗУЛЬТАТІВ

3.1. Установка основного та додаткового програмного забезпечення OMNeT++

3.1.1. Установка OMNeT++

Першим кроком для моделювання бездротової сенсорної мережі є установка обраного програмного забезпечення. Установка ПЗ проводилася у декілька кроків, установчі файли було завантажено з офіційного сайту OMNeT++, що знаходиться за адресою https://omnetpp.org/.

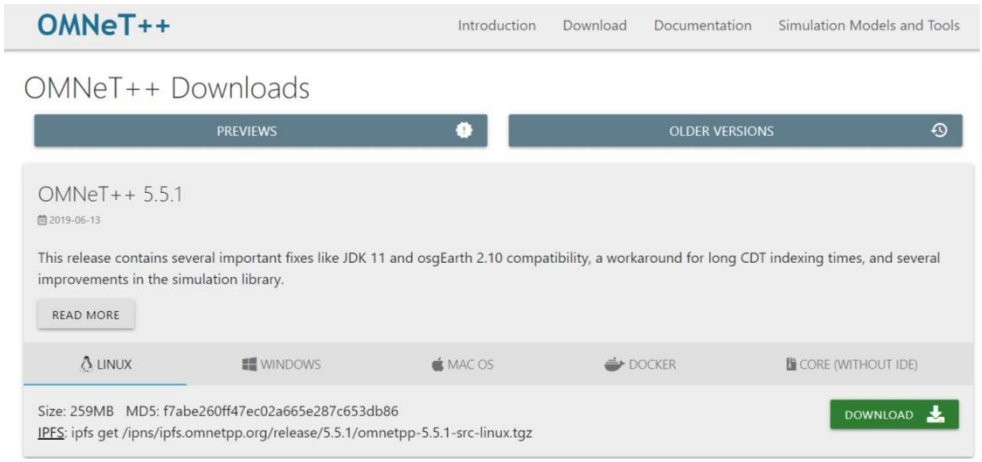


Рисунок 3.1 – Сторінка завантаження OMNeT++

Далі у командному рядку було введено наступні команди:

- ./configure
- make
- omnetpp

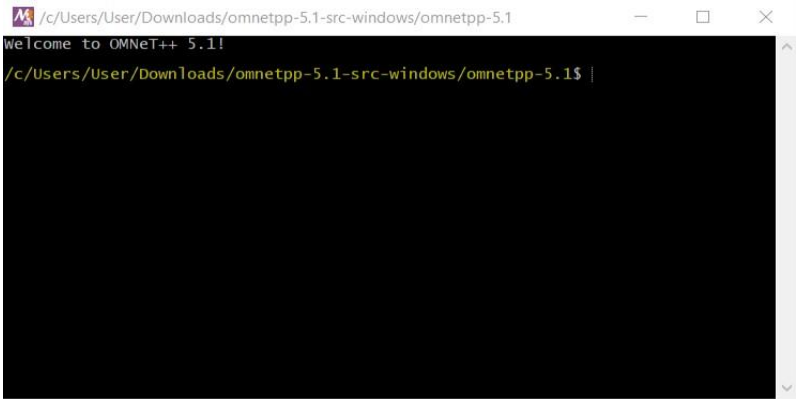


Рисунок 3.2 – Командний рядок OMNeT++

Рисунок 3.3 – Стартове вікно OMNeT++

Для подальшої роботи, за необхідності, можуть бути встановлені додатки.

3.1.2. Установка INET Framework

INET Framework – бібліотека моделей з відкритим кодом для середовища моделювання OMNeT ++. Вона надає протоколи, агенти та інші моделі для дослідників та студентів, що працюють з комунікаційними мережами. INET особливо корисний при розробці та затвердженні нових протоколів або при дослідженні нових або екзотичних сценаріїв.

INET містить моделі для Інтернет-стеку (TCP, UDP, IPv4, IPv6, OSPF, BGP тощо), протоколи дротового та бездротового зв'язку (Ethernet, PPP, IEEE 802.11 тощо), підтримка мобільності, протоколи MANET, DiffServ, MPLS з LDP та RSVP-TE, та багато інших протоколів так компонентів.

Файли для установки знаходяться за посиланням https://inet.omnetpp.org/Introduction

3.2. Моделювання

Моделювання мережі з використанням програми OMNeT++ має ряд наступних особливостей:

- Проект, створений у програмі OMNeT++, є багатофайловою ієрархічною структурою.

- Модель створюється з окремих функціональних елементів, які є частиною фреймворку INET.
- Налаштування виконується за допомогою мови C++.

Структура проекту у середовищі OMNeT++ має наступний вигляд (рис. 3.4):

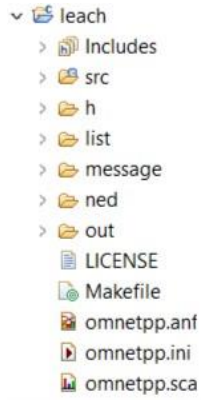


Рисунок 3.4 – Структура проекта

Проект містить у собі файли підключення, основний код, повідомлення, які передаються у межах мережі, елементи візуалізації, результуючі файли та завантажувальний файл. Структура є типовою для більшості проектів.

На рис 3.5 можна побачити схему алгоритму моделювання проекту середовищем OMNeT++.

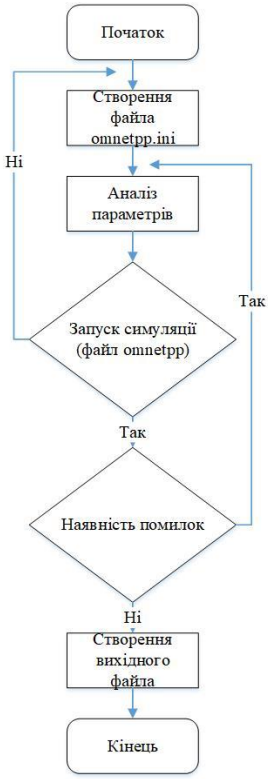


Рисунок 3.5 – Схема роботи середовища моделювання

Схема моделі, що представлена нижче (рис. 3.6), була відтворена у середовищі моделювання OMNeT++ та налаштована з використанням обраних протоколів.

3.2.1. Базова схема мережі

Для подальшого моделювання було обрано типову схему побудови мережі (рис. 3.6.)

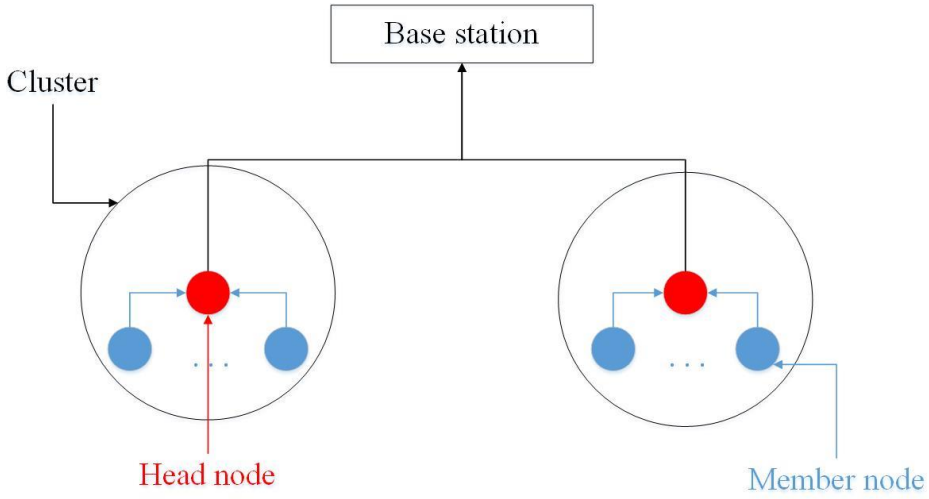


Рисунок 3.6 – Схема мережі

Дана схема має наступні компоненти:

- Базова станція (Base station), яка є регулюючою частиною мережі.
- Вузли (Nodes), які формують собою кластери. Їх кількість у побудованій моделі – 20.
- Кластери (Cluster), які працюють паралельно одне одному. Їх кількість може змінюватися у залежності від кількості сенсорних вузлів та протоколу маршрутизації, який використаний при налаштуванні. Кожен кластер містить:
    - Голова кластера (Head node), яка пов’язана безпосередньо з базовою станцією.
    - Члени кластера (Member node), які пов’язані з головою кластера.

Передача даних у мережі проводиться у межах кластерів та від голови кластера до базової станції.

Голови кластерів та члени кластерів мають одну й ту саму структуру, але, у залежності від особливостей налаштування та ходу роботи моделі, той чи інший пристрій може стати головою кластера.

Представлена схема є прийнятною для обраних протоколів і може бути застосована без значних змін в рамках експериментів, що дозволяє порівнювати роботу протоколів у приблизно однакових умовах.

За допомогою засобів обраного ПЗ та вищезазначеної схеми було змодельовано бездротову сенсорну мережу (рис. 3.7) , з параметрами, що відповідають базовій схемі. Далі представлено інформацію про налаштування даної мережі з використанням обраних протоколів.

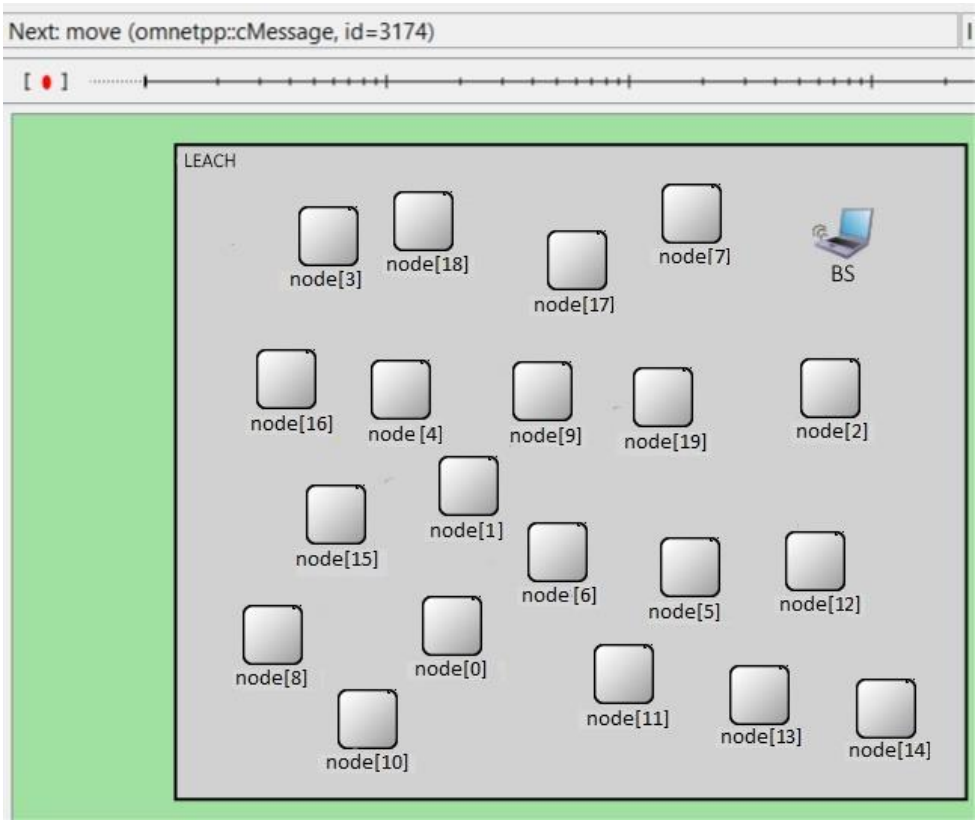


Рисунок 3.7 – Базова мережа у середовищі OMNeT++

3.2.1.1. Налаштування протоколу LEACH

Мережа була налаштована згідно до особливостей протоколу маршрутизації LEACH.

Алгоритм полягає у наступному: з використанням формули (2.1) проводиться обрання голови кластера, за умови, що число Rn менше порогового значення T. Якщо даний вузол і є головою кластера, то він анонсує свій статус іншим вузлам, після цього вузол і очікує повідомлення про приєднання до нього членів кластера. Отримавши повідомлення, він створює графік TDMA та розсилає його кожному члену.

Блок-схема, що описує роботу коду, який був написаний для коректної роботи моделі, зображена на рис. 3.7.

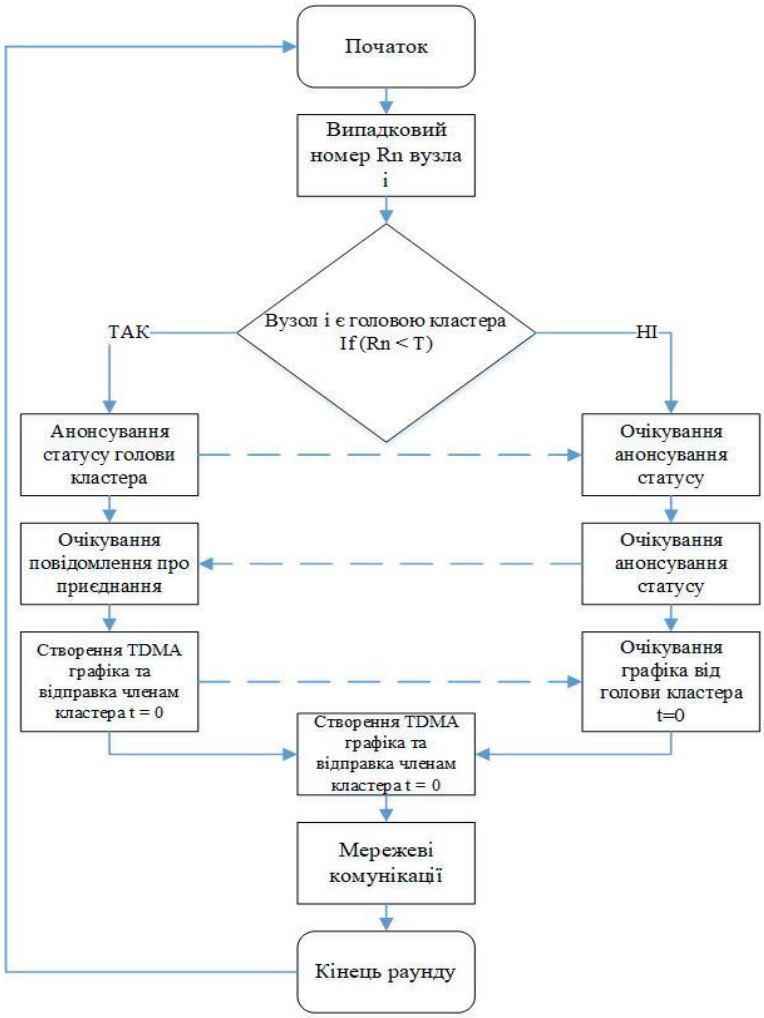


Рис. 3.7 – Блок-схема LEACH

Код знаходиться в Додатку Б.

3.2.1.2. Налаштування протоколу TEEN

Мережа була налаштована згідно до особливостей протоколу маршрутизації TEEN.

Алгоритм полягає у наступному: з використанням формули (2.1) проводиться обрання голови кластера, за умови, що число Rn менше порогового значення T. Якщо даний вузол і є головою кластера, то він анонсує свій статус іншим вузлам, після цього вузол і очікує повідомлення про приєднання до нього членів кластера. Ця частина алгоритму є ідентичною до алгоритму LEACH та має ідентичне відображення у коді. Наступним кроком є визначення атрибутів трансляції голови кластера. Ці атрибути – жорсткий поріг (HT), м’який поріг (ST) та почуття вузла. Якщо атрибут почуття вузла є першим для цього вузла та відповідає нерівності атрибут почуття < HT, то це призводить до збереження інформації та її передачі. Якщо він не є першим та відповідає нерівності атрибут почуття > HT & >= ST, то це теж призводить до

збереження та передачі інформації. В інших випадках в обох вірогідностях почуття вузла підраховується заново. Дані атрибути є фільтрами для інформації, що передається, що допомагає економити енергію.

Блок-схема, що описує роботу коду, який був написаний для коректної роботи моделі, зображена на рис. 3.8.

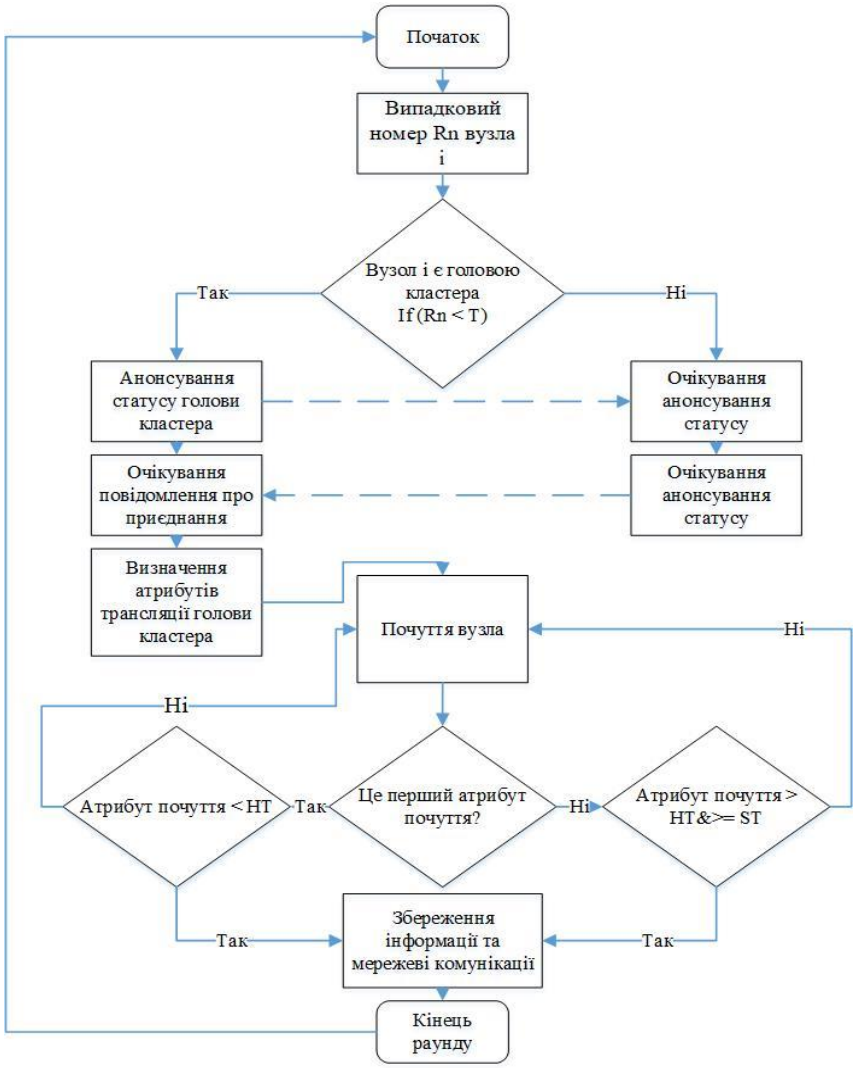


Рис. 3.8 – Блок-схема TEEN

Код знаходиться у Додатку Б.

3.2.1.3. Налаштування протоколу HEED

Мережа була налаштована згідно до особливостей протоколу маршрутизації TEEN.

Алгоритм полягає у наступному: першим кроком встановлюється міжкластерна вартість, далі за допомогою формули (2.2) кожен вузол встановлює значення ймовірності стати головою кластера. Кожен вузол перевіряється на більшовартість ймовірності. Головою кластера стає той

вузол, який має найбільшу ймовірність. Члени вузлів приєднуються до найближчих голів кластера, що призводить до налагодження міжкластерної комунікації.

Блок-схема, що описує роботу коду, який був написаний для коректної роботи моделі, зображена на рис. 3.9.

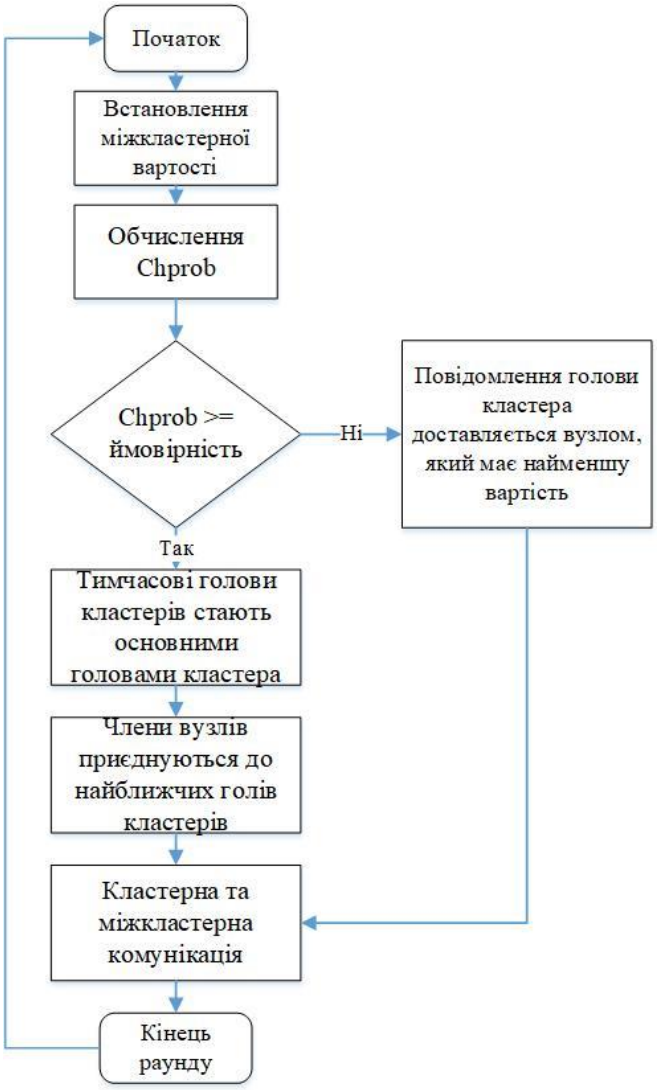


Рис. 3.9 – Блок-схема HEED

Код знаходиться у Додатку Б.

3.3. Проведення експерименту

Для порівняння роботи отриманих моделей було вирішено провести експеримент, який дозволив у повному обсязі протестувати особливості роботи протоколів стосовно одне одного.

Експеримент полягає у наступному: було проведено ряд перевірок роботи кожної моделі за конкретними параметрами. Обрані параметри: затримки у мережі (мінімальна та

максимальна), стандартне відхилення у вузлах мережі, передача пакетів у мережі. Після цього було проведено порівняльний аналіз отриманих результатів.

3.3.1. Перший параметр: затримка у вузлах мережі

Параметр затримки (delay, latency) – один із найважливіших у функціонуванні мережі, саме його показники відповідають за те, через скільки пакет відправлений з точки А виявитися в точці Б.

У середовищі OMNeT++ можна отримати показники мінімальної та максимальної затримки. Файл з результатами моделювання знаходиться таким шляхом: File > New > Analysis File.

Таблиця (табл. 3.1), представлена нижче, містить інформацію про затримки для моделей з протоколами LEACH, TEEN, HEED.

Таблиця 3.1 – Затримки у вузлах мережі

	LEACH		TEEN		HEED	
	max	min	max	min	max	min
node 0	2.8	0,8	3,1	0,5	1,8	0,2
node 1	0	0	2,1	0,2	1	0,5
node 2	4.3	0,2	2,2	0,3	2,1	0,3
node 3	0	0	1,1	0,5	1,3	0,4
node 4	5.1	0,7	3,2	0,6	1,2	0,4
node 5	3,1	0,3	0	0	1,4	0,4
node 6	4,6	0,5	2,3	0,3	1,7	0,5
node 7	3,1	0,8	1,4	0,4	2	0,3
node 8	0	0	2,1	0,4	1,1	0,2
node 9	5,1	0,4	3,8	0,1	2,1	0,3
node 10	3,1	0,1	1,5	0,3	1,8	0,1
node 11	5,5	0,6	2,5	0,5	3,2	0,6
node 12	4,9	0,9	2,4	0,1	1,2	0,3
node 13	5,4	0,8	0	0	1,6	0,4
node 14	0	0	1,6	0,5	2,3	0,1
node 15	6,1	1,5	3,7	0,4	3,3	0,1
node 16	3,7	0,6	2,3	0,3	2,4	0,1

Продовження таблиці 3.1

node 17	4,2	0,5	2,2	0,1	1,8	0,3
node 18	0	0	1,3	0,1	1,1	0,3
node 19	6,2	1,1	3,9	0,9	2,2	0,6

Для більшої наочності далі представлені скріншоти програми, які являють собою графіки мінімальних та максимальних затримок.

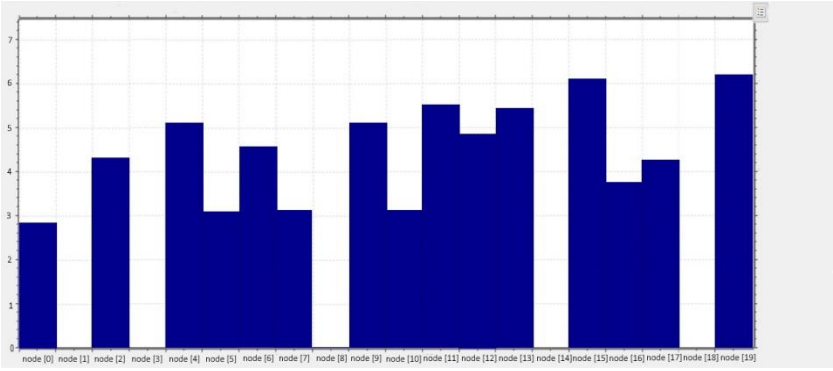


Рисунок 3.11 – Графік максимальних затримок LEACH

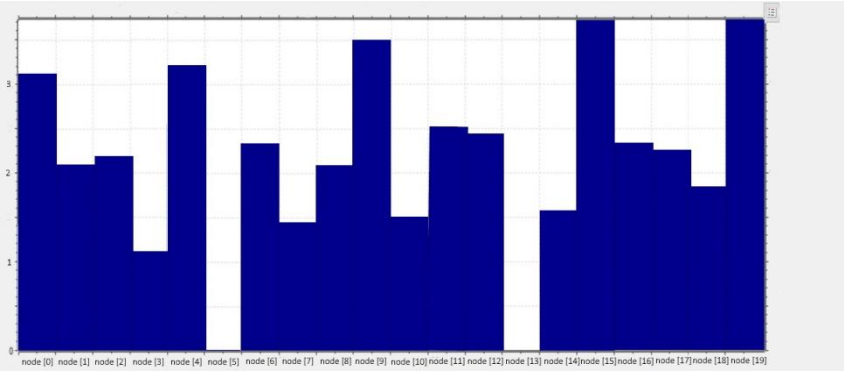


Рисунок 3.12 – Графік максимальних затримок TEEN

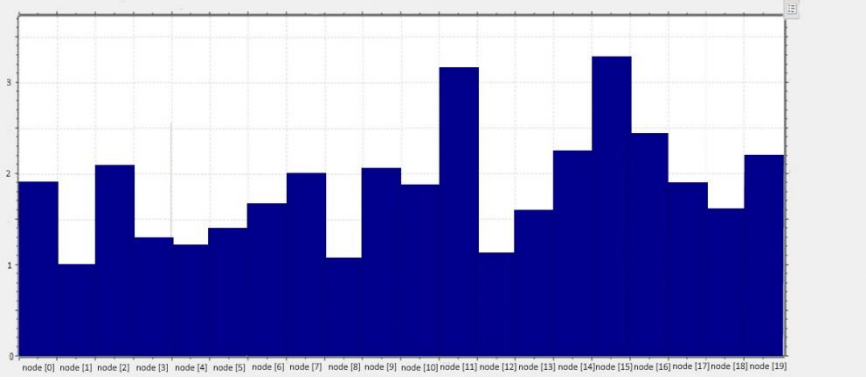


Рисунок 3.13 – Графік максимальних затримок HEED

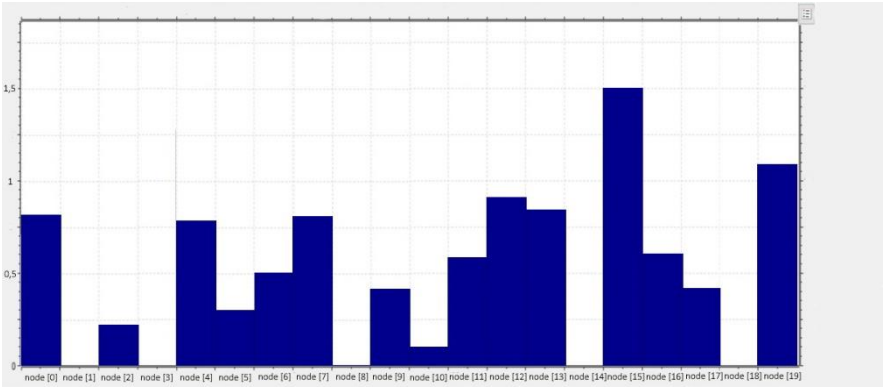


Рисунок 3.14 – Графік мінімальних затримок LEACH

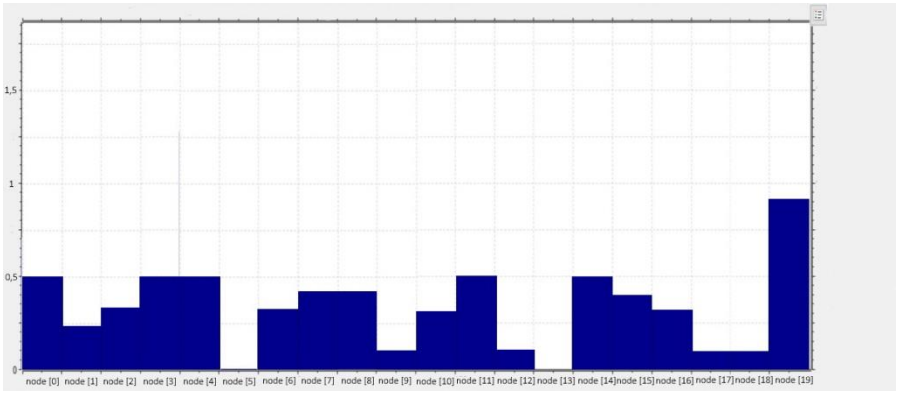


Рисунок 3.15 – Графік мінімальних затримок TEEN

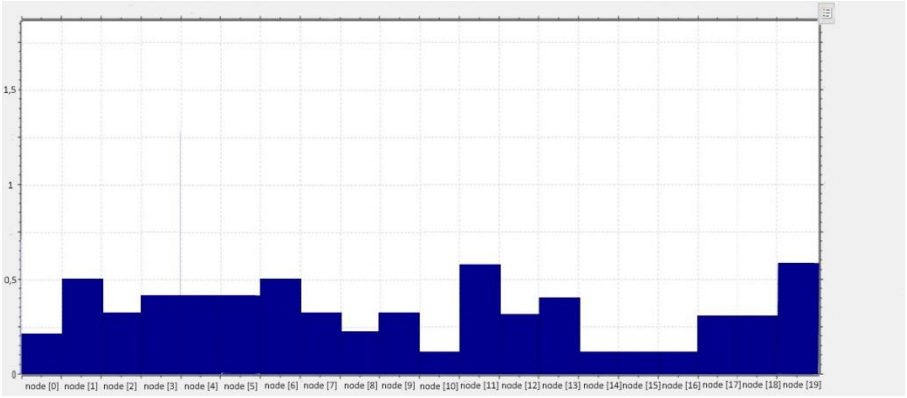


Рисунок 3.16 – Графік мінімальних затримок HEED

3.3.2. Другий параметр: стандартне відхилення у вузлах мережі

Параметр стандартного відхилення (standart deviation, stddev) показує розрізнення значень випадкової величини відносно її середніх значень. Чим більша різниця, тим складніше керувати трафіком прийняття пакетів в потрібному порядку, уникання дублювання пакетів.

Таблиця (табл. 3.2), представлена нижче, містить інформацію про стандартне відхилення для моделей з протоколами LEACH, TEEN, HEED.

Таблиця 3.2 – Стандартне відхилення у вузлах мережі

	LEACH	TEEN	HEED
node 0	1,5	0,7	0,1
node 1	0	1,2	0,2
node 2	0,6	0,4	0,3
node 3	0	0,9	0,4
node 4	1,2	0,9	0,1
node 5	1,9	0	0,4
node 6	0,6	1,1	0,1
node 7	1,5	1	0,1
node 8	0	0,5	0,4
node 9	0,5	1,3	0,3
node 10	1,2	0,5	0,2
node 11	0,6	0,6	0,2
node 12	0,2	1,4	0,2
node 13	1,7	0	0,3
node 14	0	0,5	0,5
node 15	1,8	0,7	0,3
node 16	1	0,2	0,3
node 17	1,1	0,4	0,4
node 18	0	0,2	0,2
node 19	2	1,4	0,5

Далі представлені скріншоти, що візуалізують отримані дані.

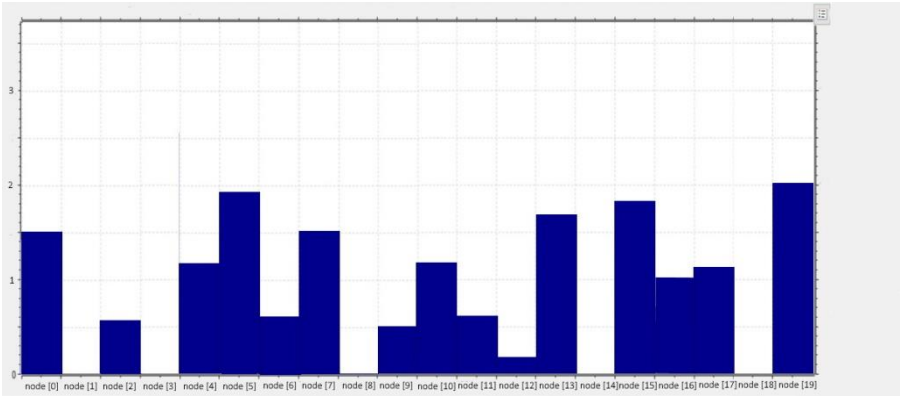


Рисунок 3.17 – Графік відхилень LEACH

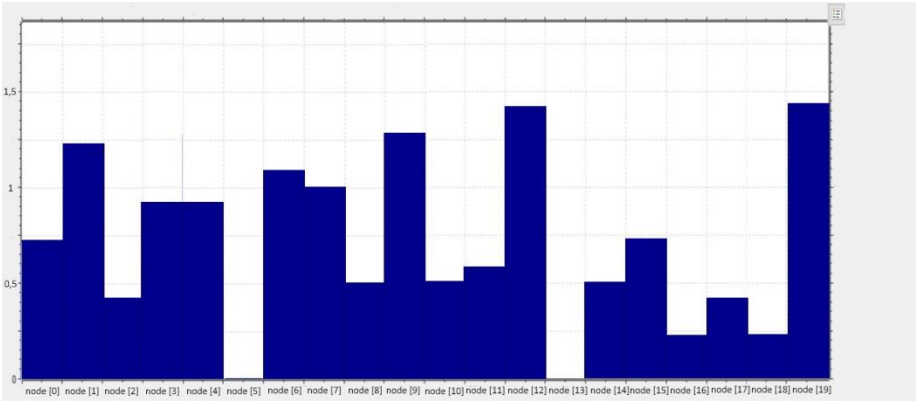


Рисунок 3.18 – Графік відхилень TEEN

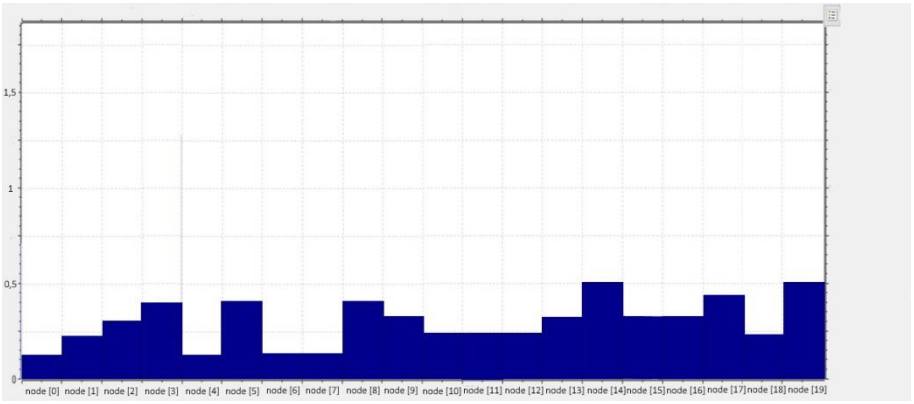


Рисунок 3.19 – Графік відхилень HEED

3.3.3. Третій параметр: передача даних у мережі

Параметр передачі пакетів (packets received) дає уявлення про стабільність та працездатність мережі у реальних умовах. Зважаючи на те, що кожен з вузлів у той чи інший час має можливість стати головою кластера, кожен вузол як приймає, так і передає пакети даних.

Таблиця (табл. 3.3), представлена нижче, містить інформацію про передачу даних для моделей з протоколами LEACH, TEEN, HEED.

Таблиця 3.3 – Передача даних у мережі

	LEACH	TEEN	HEED
node 0	29	20	19
node 1	0	12	35
node 2	14	22	21

Продовження таблиці 3.3

node 3	0	31	16
node 4	28	18	17
node 5	11	0	23
node 6	21	17	36
node 7	31	16	14
node 8	0	16	28
node 9	33	16	30
node 10	12	10	13
node 11	18	9	15
node 12	17	12	18
node 13	13	0	24
node 14	0	27	24
node 15	27	27	22
node 16	22	36	33
node 17	25	14	22
node 18	0	14	21
node 19	8	10	8

Нижче представлені графіки передачі пакетів у вузлах.

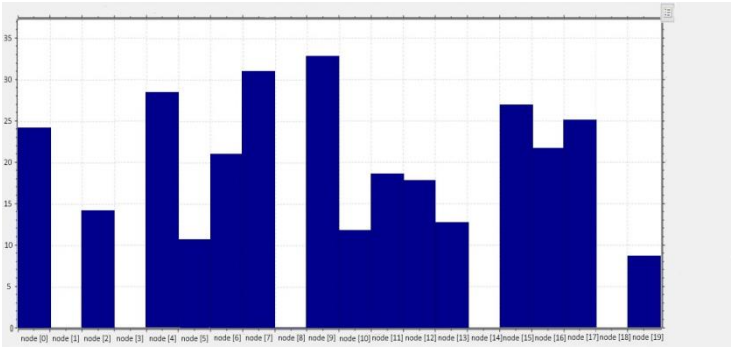


Рисунок 3.20 – Графік передачі пакетів LEACH

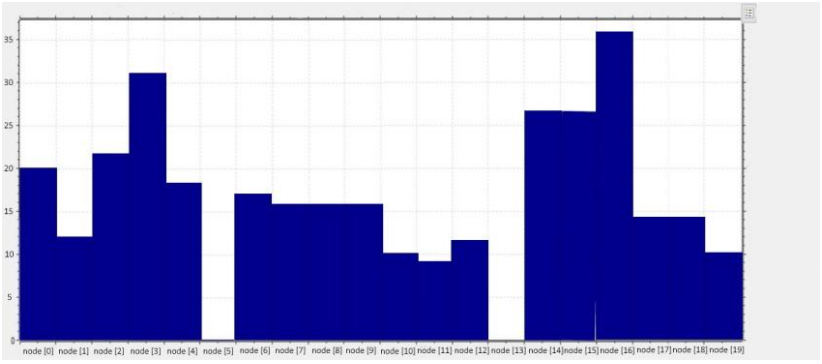


Рисунок 3.21 – Графік передачі пакетів TEEN

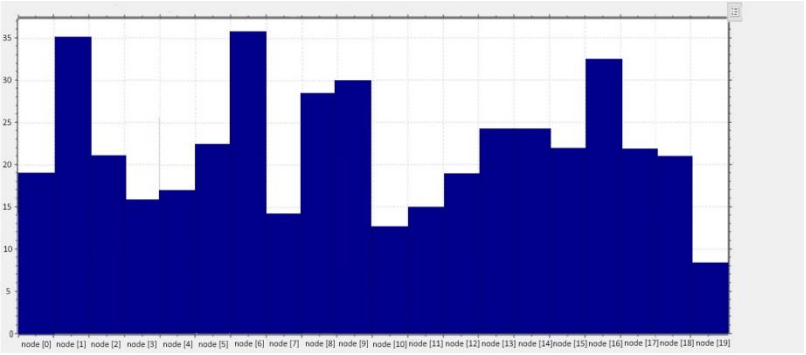


Рисунок 3.22 – Графік передачі пакетів HEED

3.4. Порівняльний аналіз результатів

Для формулювання об’єктивних висновків щодо отриманих результатів було створено суміщені графіки, які демонструють різницю у показниках стосовно кожного з параметрів.

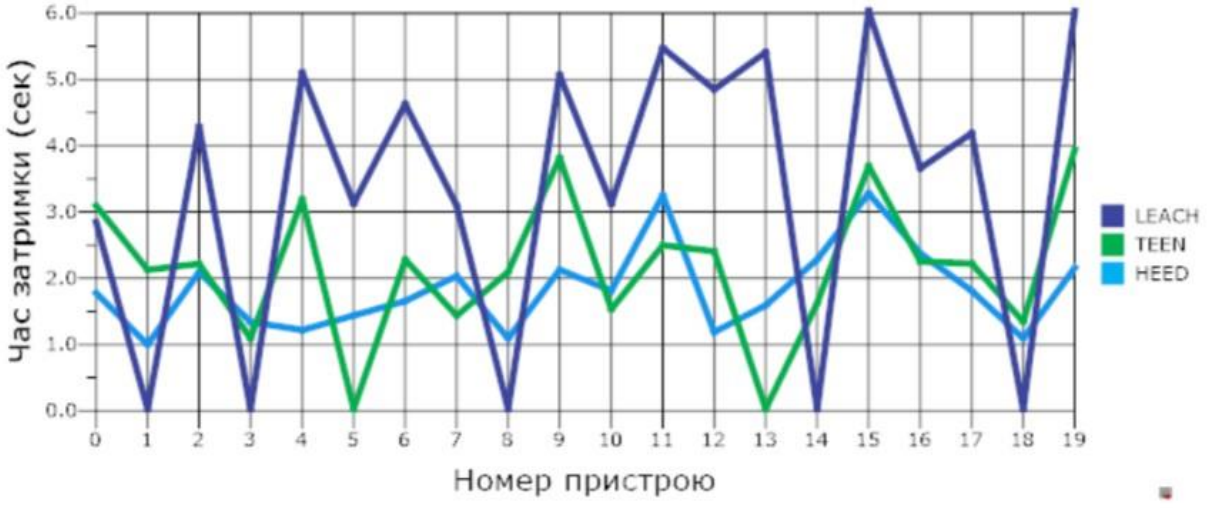


Рисунок 3.23 – Графік співвідношення максимальної затримки

Дивлячись на графік можна зробити висновок, що найбільші затримки були відмічені у моделі, яка була налаштована за допомогою протоколу LEACH, затримки у моделях з протоколами TEEN та HEED не мають настільки істотних відмінностей. Вузли 1, 3, 8, 14 та 18 у моделі з протоколом LEACH не мають затримки, що найвірогідніше свідчить про те, що інформація з них не була передана далі. У той час як модель з протоколом TEEN має подібні показники лише у вузлах 5 та 13, а модель з протоколом HEED не має взагалі.

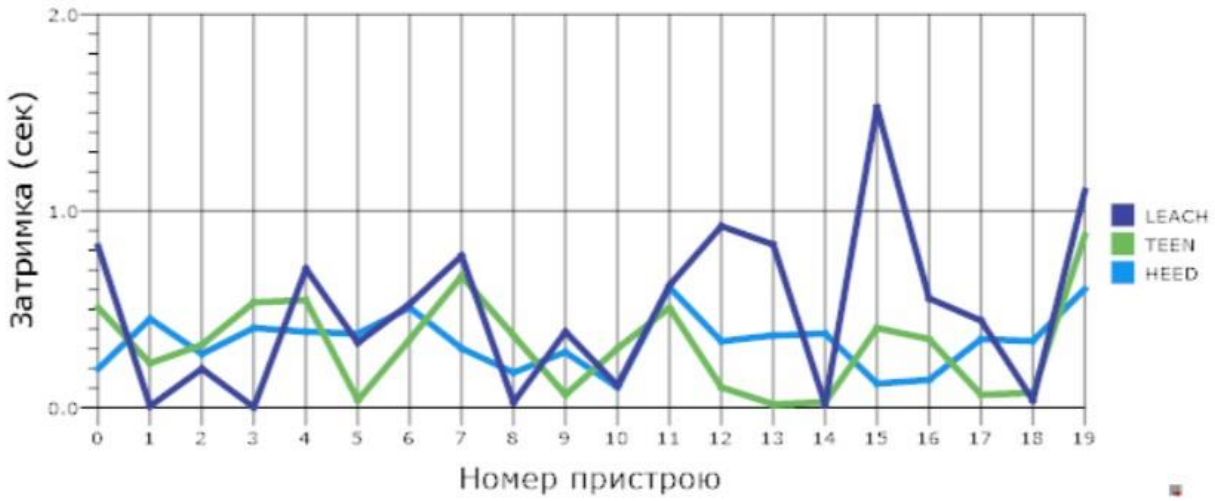


Рисунок 3.24 – Графік співвідношення мінімальної затримки

Графік мінімальних затримок підтвердив вищезазначені показники у повному обсязі. Протоколи TEEN та HEED також не мають істотних відмінностей, що робить їх конкурентоздатними стосовно одне одного.

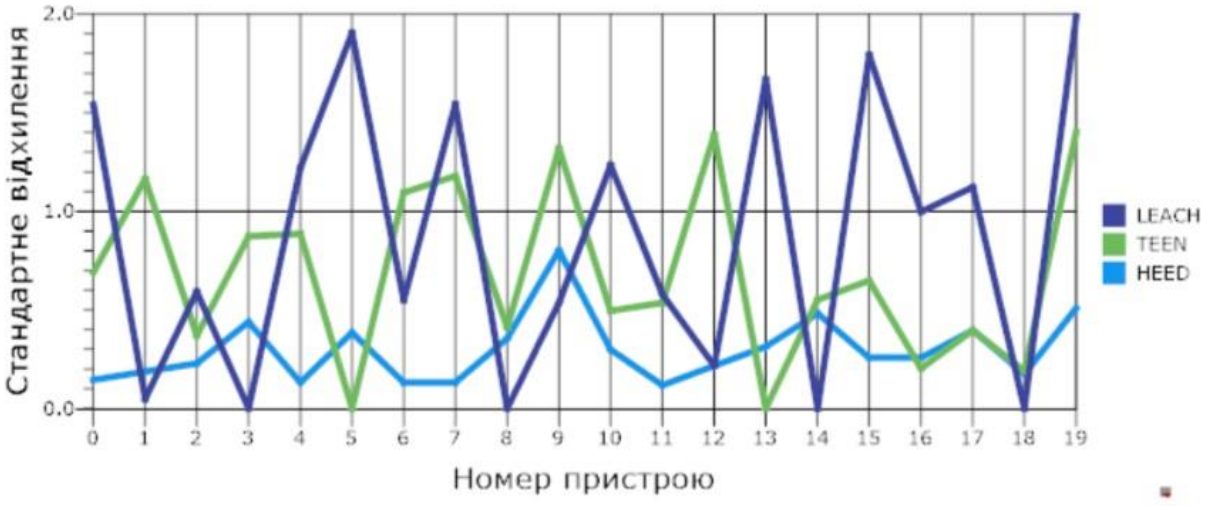


Рисунок 3.25 – Графік співвідношення стандартного відхилення

Стандартне відхилення у моделях сильно розрізняється. Найменше відхилення та найбільшу стійкість до переривань демонструє протокол HEED, найбільше – LEACH. Вузли 1, 3, 8, 14, 18 у моделі LEACH; 5, 13 у моделі TEEN мають відхилення 100%.

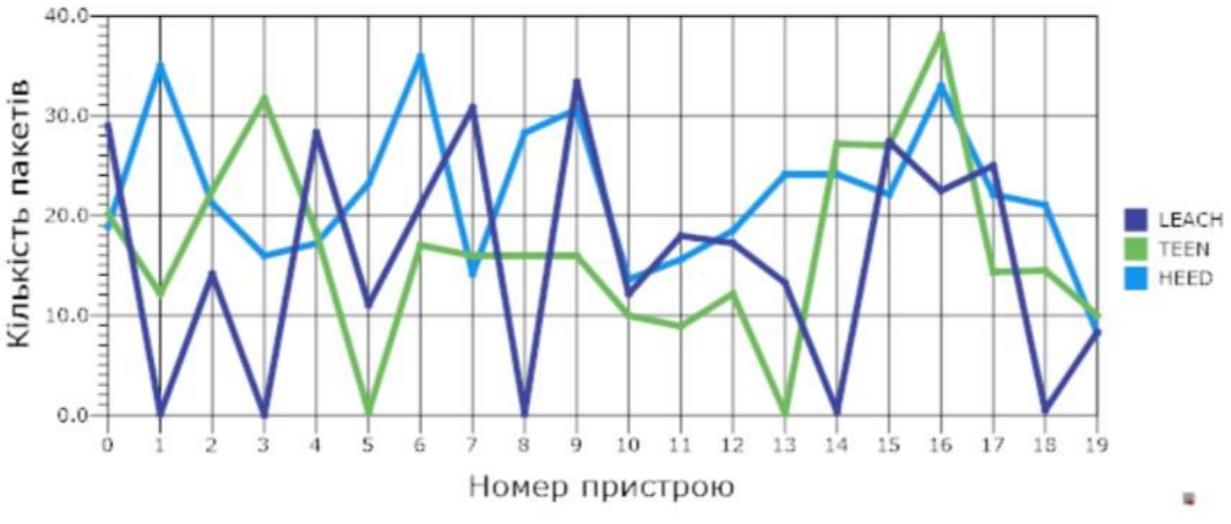


Рисунок 3.26 – Графік співвідношення кількості пакетів

Кількість переданих пакетів на пікових значеннях у всіх трьох моделях є близькою одне до одної, але переривання призвело до відсутності передачі пакетів у п’яти вузлах моделі з протоколом LEACH, та двох вузлах моделі з протоколом TEEN.

Таким чином, протокол HEED виявився більш стійким до переривань, що робить його найбільш прийнятним до використання у реальних умовах.

3.5. Математичний аналіз

Для підтвердження отриманих даних було проведено математичний аналіз отриманих результатів для кожної моделі. Параметром аналізу були обрані затримки у вузлах мережі та формула критерію сумарної середньої затримки (1.1). Для цього було розраховано середні значення максимальної та мінімальної затримки, проведено розрахунок за вищезазначеною формулою та порівняно результати. Згідно до формули, найкращим результатом розрахунку є мінімально можливий результат. Розрахунки представлені нижче:

Модель з використанням протоколу LEACH:

$$(2.8+0+4.3+0+5.1+3.1+4.6+3.1+0+5.1+3.1+5.5+4.9+5.4+0+6.1+ 3.7+4.2+0+6.2)/20=3.36,$$

$$(0.8+.+0.2+0+0.7+0.3+0.5+0.8+0+0.4+0.1+0.6+0.9+0.8+1.5+ 0.6+0.6+0+1.1)/20=0.49,$$

$$\sum_{i=1}^{20}(3.36+0.49)=77$$

Модель з використанням протоколу TEEN:

$(3.1+2.1+2.2+1.1+3.2+0+2.3+1.4+2.1+3.8+1.5+2.5+2.4+0+1.6+3.7+$
$2.3+2.2+1.3+3.9)/20=2.135,$

$(0.5+0.2+0.3+0.5+0.6+0+0.3+0.4+0.1+0.3+0.5+0.1+0+0.5+0.4+$
$0.3+0.1+0.1+0.9)/20=0.325,$

$$\sum_{i=1}^{20}(2.135+0.325)=49.2$$

Модель з використанням протоколу HEED:

$(1.8+1+2.1+1.3+1.2+1.4+1.7+2+1.1+2.1+1.8+3.2+1.2+1.6+2.3+3.3+$
$2.4+1.8+1.1+2.2)/20=1.83,$

$(0.2+0.5+0.3+0.4+0.4+0.4+0.5+0.3+0.2+0.3+0.1+0.6+0.3+0.4+0.1+$
$0.1+0.1+0.3+0.3+0.6)/20=0.32,$

$$\sum_{i=1}^{20}(1.83+0.32)=43$$

Найменше значення коефіцієнту – 43. Таким чином, протокол HEED є найбільш придатним до використання, математичний аналіз цілком підтвердив отримані результати.

Висновки до третього розділу

Практично перевірено функціональність середовища моделювання OMNeT++, проаналізована його внутрішня структура та особливості побудови проектів. Для цього було створено модель бездротової сенсорної мережі, яка була налаштована за допомогою заздалегідь обраних протоколів маршрутизації LEACH, TEEN та HEED, після чого було проведено ряд перевірок працездатності отриманих моделей.

Середовище моделювання OMNeT++ дозволяє створювати реалістичні моделі бездротових сенсорних мереж та аналізувати їх стан за багатьма необхідними параметрами, що і було проведено з отриманими моделями. Моделі були перевірені за наступними критеріями:

- затримки у вузлах мережі;
- стандартне відхилення у вузлах мережі;
- передача пакетів у вузлах мережі.

Моделі було проаналізовані за допомогою математичних моделей.

Обраний найбільш вигідний з точки зору практичного застосування протокол маршрутизації – HEED.

Сформулювано висновки щодо середовища моделювання OMNeT++, згідно зякими середовище є повністю відповідним до вимог, які було висунуто до подібного ПЗ.

РОЗДІЛ 4
ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

В даному розділі проведено аналіз потенційних небезпечних та шкідливих виробничих факторів, причин пожеж. Розглянуті заходи, які дозволяють забезпечити гігієну праці і виробничу санітарію. На підставі аналізу розроблені заходи з техніки безпеки та рекомендації з пожежної профілактики.

Завданням даної магістерської роботи було: дослідження роботи та моделювання бездротових сенсорних мереж у контексті IoT.

4.1 Аналіз стану умов праці

4.1.1 Вимоги до приміщень

Геометричні розміри приміщення зазначені в табл. 4.1.

Таблиця 4.1 - Розміри приміщення.

Найменування	Значення
Довжина, м	4,4
Ширина, м	2,8
Висота, м	2,5
Площа, $м^2$	12,32
Об’єм, $м^3$	30,8

Згідно з ДСН 3.3.6.042-99 [1] розмір площі для одного робочого місця оператора персонального комп’ютера має бути не менше 6 кв. м, а об’єм — не менше 20 куб. м. Отже, дане приміщення цілком відповідає зазначеним нормам.

Для забезпечення потрібного рівного освітленості кімната має вікно та систему загального рівномірного освітлення, що встановлена на стелі. Для дотримання вимог пожежної безпеки встановлено порошковий вогнегасник та систему автоматичної пожежної сигналізації.

4.1.2 Вимоги до організації місця праці

При порівнянні відповідності характеристик робочого місця нормативним основні вимоги до організації робочого місця за ДСанПіН 3.3.2.007-98 [2] (табл. 4.2) і відповідними фактичними значеннями для робочого місця, констатуємо повну відповідність.

Таблиця 4.2 - Характеристики робочого місця

Найменування параметра	Фактичне Значення	Нормативне Значення
Висота робочої поверхні, мм	700	680 ÷ 800
Висота простору для ніг, мм	650	не менше 600
Ширина простору для ніг, мм	540	не менше 500
Глибина простору для ніг, мм	660	не менше 650
Висота поверхні сидіння, мм	420	400 ÷ 500
Ширина сидіння, мм	410	не менше 400
Глибина сидіння, мм	420	не менше 400
Висота поверхні спинки, мм	500	не менше 300
Ширина опорної поверхні спинки, мм	400	не менше 380
Радіус кривини спинки в горизонтальній площині, мм	400	400
Відстань від очей до екрану дисплея, мм	750	700 ÷ 800

4.2 Навантаження та напруженість процесу праці

Під час виконання робіт використовують ПК та периферійні пристрої, що призводить до навантаження на окремі системи організму. Такі перекоси у напруженні різних систем організму, що трапляються під час роботи з ПК, зокрема, значна напруженість зорового аналізатора і довготривале малорухоме положення перед екраном, не тільки не зменшують загального напруження, а навпаки, призводять до його посилення і появи стресових реакцій.

Найбільшому ризику виникнення різноманітних порушень піддаються: органи зору, м’язово-скелетна система, нервово-психічна діяльність, репродуктивна функція у жінок.

Тобто наявне психофізіологічні небезпечні та шкідливі фактори:

а) фізичного перевантаження:

- статичного;
- динамічного;

б) нервово-психічного перевантаження:

- розумового перенапруження;
- монотонності праці;
- перенапруження аналізаторів;
- емоційних перевантажень.

Роботу за дипломним проектом визнано, таку, що займає 50% часу робочого дня та за восьмигодинної робочої зміни рекомендовано встановити додаткові регламентовані перерви:

- для розробників програм тривалістю 15 хв через кожну годину роботи.

4.3 Виробнича санітарія

На підставі аналізу небезпечних та шкідливих факторів при виробництві (експлуатації), пожежної безпеки можуть бути надалі вирішені питання необхідності забезпечення працюючих достатньою кількістю освітлення, вентиляції повітря, організації заземлення, тощо.

4.3.1 Аналіз небезпечних та шкідливих факторів при виробництві (експлуатації) виробу

Аналіз небезпечних та шкідливих виробничих факторів виконується у табличній формі (табл. 4.3). Роботу, пов'язану з ЕОП з ВДТ, у тому числі на тих, які мають робочі місця, обладнані ЕОМ з ВДТ і ПП, виконують із забезпеченням виконання НПАОП 0.00-7.15-18 [5], яке встановлюють вимоги безпеки до обладнання робочих місць, до роботи із застосуванням ЕОМ з ВДТ і ПП. Переважно роботи за проектами виконують у кабінетах чи інших приміщеннях, де використовують різноманітне електрообладнання, зокрема персональні комп’ютери (ПК) та периферійні пристрої. Основними робочими характеристиками персонального комп'ютера є:

– робоча напруга U=+220В +-5%;

– робочий струм I=2A;

– споживана потужність P=350 Вт.

Таблиця 4.3 – Аналіз небезпечних і шкідливих виробничих факторів

Небезпечні і шкідливі виробничі фактори	Джерела факторів (види робіт)	Кількісна Оцінка	Нормативні Документи
1	2	3	4

Продовження таблиці 4.3

Фізичні:			
підвищена або знижена вологість повітря	-//-	3	[1]
підвищений рівень напруги електричної мережі, замикання якої може відбутися через тіло людини	-//-	3	[3] [4]
Психофізіологічні:			
нервово-психічна перевантаження (розумове, перенапруження аналізаторів-зорових)	- формулювання теми; - пошук інформацію про предметну область; - пошук інформації про наявні аналоги; - проектування структур та алгоритмів; - виконання роботи; - оформлення записки.	4	[4] [5]
фізичні (статичне - сидіння)	порушення умов організації робочого часу (безперервна робота)	2	[2] [5]

4.3.2 Пожежна безпека

Висока щільність елементів в електронних схемах призводить до значного підвищення температури окремих вузлів (80…100 ℃). При проходженні електричного струму по провідниках і деталей виділяється тепло, що в умовах їх високої щільності може привести до перегріву, і може служити причиною запалювання ізоляційних матеріалів. Слабкий опір ізоляційних матеріалів дії температури може викликати порушення ізоляції і привести до короткого замикання між струмоведучими частинами обладнання (шини, електроди).

Заземлені конструкції, що знаходяться в приміщеннях, де розміщені робочі місця (батареї опалення, водопровідні труби, кабелі із заземленим відкритим екраном), надійно захищені діелектричними щитками та/або сітками з метою недопущення потрапляння працівника під напругу.

В приміщенні наявна затверджена «План-схема евакуації з кабінету (приміщення)».

Горючими матеріалами в приміщенні, де розташовані ЕОМ, є:

1) поліамід - матеріал корпусу мікросхем, горюча речовина, температура самозаймання 420℃;

2) полівінілхлорид - ізоляційний матеріал, горюча речовина, температура запалювання 335℃, температура самозаймання 530℃;

3) склотекстоліт ДЦ - матеріал друкарських плат, важкогорючий матеріал, показник горючості 1.74, не схильний до температурного самозаймання;

4) пластикат кабельний №489 - матеріал ізоляції кабелів, горючий матеріал, показник горючості більше 2.1;

5) деревина - будівельний і обробний матеріал, з якого виготовлені меблі, горючий матеріал, показник горючості більше 2.1, температура запалювання 255℃, температура самозаймання 399℃.

Простори усередині приміщень в межах, яких можуть утворюватися або знаходиться пожежонебезпечні речовини і матеріали відповідно до ДСТУ Б В.1.1-36:2016 [6] відносяться до пожежонебезпечної зони класу П-IIа. Це обумовлено тим, що в приміщенні знаходяться тверді горючі та важкозаймисті речовини та матеріали. Приміщенню, у якому розташоване робоче місце, присвоюється II ступень вогнестійкості.

Причинами можливого загоряння і пожежі можуть бути:

1) несправність електроустановки;

2) конструктивні недоліки устаткування;

3) коротке замикання в електричних мережах;

4) запалювання горючих матеріалів, що знаходяться в безпосередній близькості від електроустановки.

Продуктами згорання, що виділяються на пожежі, є: окис вуглецю; сірчистий газ; окис азоту; синильна кислота; акромін; фосген; хлор і ін. При горінні пластмас, окрім звичних продуктів згорання, виділяються різні продукти термічного розкладання: хлорангідридні кислоти, формальдегіди, хлористий водень, фосген, синильна кислота, аміак, фенол, ацетон, стирол.

4.3.3 Електробезпека

На робочому місці виконуються наступні вимоги електробезпеки: ПК, периферійні пристрої та устаткування для обслуговування, електропроводи і кабелі за виконанням та ступенем захисту відповідають класу зони за ПУЕ, мають апаратуру захисту від струму короткого замикання та інших аварійних режимів. Лінія електромережі для живлення ПК, периферійних пристроїв і устаткування для обслуговування, виконана як окрема групова три- провідна мережа, шляхом прокладання фазового, нульового робочого та нульового захисного провідників. Нульовий захисний провідник використовується для заземлення електроприймачів. Штепсельні з'єднання та електророзетки крім контактів фазового та нульового робочого провідників мають

спеціальні контакти для підключення нульового захисного провідника. Електромережа штепсельних розеток для живлення персональних ПК, укладено по підлозі поруч зі стінами відповідно до затвердженого плану розміщення обладнання та технічних характеристик обладнання. Металеві труби та гнучкі металеві рукави заземлені. Захисне заземлення включає в себе заземлюючих пристроїв і провідник, який з'єднує заземлюючий пристрій з обладнанням, яке заземлюється - заземлюючий провідник.

4.4 Гігієнічні вимоги до параметрів виробничого середовища

4.4.1 Мікроклімат

Мікроклімат робочих приміщень - це клімат внутрішнього середовища цих приміщень, що визначається діючої на організм людини з'єднанням температури, вологості, швидкості переміщення повітря. В даному приміщенні проводяться роботи, що виконуються сидячи і не потребують динамічного фізичного напруження, то для нього відповідає категорія робіт 1а. Отже оптимальні значення для температури, відносної вологості й рухливості повітря для зазначеного робочого місця відповідають ДСН 3.3.6.042-99 [1] і наведені в табл. А.4:

Таблиця 4.4 – Норми мікроклімату робочої зони об’єкту

Період Року	Категорія Робіт	Температура C^0	Відносна вологість %	Швидкість руху повітря, м/с
Холодна	Легка-1а	22-24	40-60	0,1
Тепла	Легка-1а	23-25	40-60	0,1

4.4.2 Освітлення

Для виробничих та адміністративних приміщень світловий коефіцієнт приймається не менше -1/8, в побутових - 1/10:

$$S_b = \left(\frac{1}{5} / \frac{1}{10}\right) * S_n \qquad (4.1)$$

де S_b – площа віконних прорізів, $м^2$;

S_n – площа підлоги, $м^2$.

$S_n = a \cdot b = 4{,}4 \cdot 2{,}8 = 12{,}32$ $м^2$,

$S = 1/10 \cdot 25 = 1{,}232$ $м^2$.

Приймаємо 1 вікно площею $S=1{,}6$ $м^2$.

Світильники загального освітлення розташовуються над робочими поверхнями в рівномірно-прямокутному порядку. Для організації освітлення в темний час доби

передбачається обладнати приміщення, довжина якого складає 4,4 м, ширина 2,8 м, світильниками ЛПО2П, оснащеними лампою типа ЛБ (одна - 80 Вт) з світловим потоком 5400 лм. Розрахунок штучного освітлення виробляється по коефіцієнтах використання світлового потоку, яким визначається потік, необхідний для створення заданої освітленості при загальному рівномірному освітленні. Розрахунок кількості світильників n виробляється по формулі (А.2):

$$n = \frac{E * S * Z * K}{F * U * M} \qquad (4.2)$$

де Е - нормована освітленість робочої поверхні, визначається нормами – 300 лк;

S - освітлювана площа, м^2 ; S = 12,32 м^2 ;

Z - поправочний коефіцієнт світильника (Z = 1,15 для ламп розжарювання та ДРЛ; Z = 1,1 для люмінесцентних ламп) приймаємо рівним 1,1;

К - коефіцієнт запасу, що враховує зниження освітленості в процесі експлуатації – 1,5;

U - коефіцієнт використання, залежний від типу світильника, показника індексу приміщення і т.п.

- 0,575 М - число люмінесцентних ламп в світильнику - 1;

F - світловий потік лампи - 5400лм (для ЛБ-80).

Підставивши числові значення у формулу (4.2), отримуємо:

$$n = \frac{300 * 12{,}32 * 1{,}15 * 1{,}5}{5400 * 0{,}575 * 1} \approx 2{,}0$$

Приймаємо освітлювальну установку, яка складається з 2-х світильників, оснащених лампами типа ЛБ (одна - 80 Вт) зі світловим потоком 5400 лм.

4.5 Вентилювання

У приміщенні, де знаходяться ЕОМ, повітрообмін реалізується за допомогою природної організованої вентиляції (вентиляційні шахти) і установки в віконному отворі автономного кондиціонера БК-2000. Цей метод забезпечує приток потрібної кількості свіжого повітря (30 м^3 на годину на одного працюючого).

Також має здійснюватися провітрювання приміщення, в залежності від погодних умов, тривалість повинна бути не менше 10 хв. Найкращий обмін повітря здійснюється при наскрізному провітрюванні.

4.6 Заходи з організації виробничого середовища та попередження виникнення надзвичайних ситуацій

Відповідно до санітарно-гігієнічних нормативів та правил експлуатації обладнання наводимо приклади деяких заходів безпеки.

1) Заходи безпеки під час експлуатації персонального комп'ютера та периферійних пристроїв передбачають:

- правильне організування місця праці та дотримання оптимальних режимів праці та відпочинку під час роботи з ПК;

- експлуатацію сертифікованого обладнання;

- дотримання заходів електробезпеки;

- забезпечення оптимальних параметрів мікроклімату;

- забезпечення раціонального освітлення місця праці (освітленість робочого місця не перевищувала 2/3 нормальної освітленості приміщення);

- облаштовуючи приміщення для роботи з ПК, потрібно передбачити припливно-витяжну вентиляцію або кондиціювання повітря:

а) якщо об'єм приміщення 20 м^3, то потрібно подати не менш як 30 м^3/год повітря;

б) якщо об'єм приміщення у межах від 20 до 40 м^3, то потрібно подати не менш як 20 м^3/год повітря;

в) якщо об'єм приміщення становить понад 40 м^3, допускається природна вентиляція, у випадку, коли немає виділення шкідливих речовин.

- зниження рівня шуму та вібрації:

а) у джерелі виникнення, шляхом застосування раціональних конструкцій, нових матеріалів і технологічних процесів;

б) звукоізолювання устаткування за допомогою глушників, резонаторів, кожухів, захисних конструкцій, оздоблення стін, стелі, підлоги тощо;

в) використання засобів індивідуального захисту).

Розрахунок захисного заземлення (забезпечення електробезпеки будівлі).

Загальний опір захисного заземлення визначається за формулою:

$$R_{ззп} = \frac{R_з * R_п}{R_п * n * \eta_з + R_з * n_п} \qquad (4.3)$$

де Rз - опір заземлення, якими когут бать труби, опори, кути і т.п., Ом;

Rш - опір опори, яке з'єднує заземлювачі, Ом;

п - кількість заземлювачів;

ηз - коефіцієнт екранування заземлювача; приймається в межах 0,2 ÷ 0,9; ηз = 0,7

ηш - коефіцієнт екранування сполучної стійки; приймається в межах 0,1 ÷ 0,7; ηш = 0,5;

Опір заземлення визначається за формулою:

$$R_з = \frac{\rho}{2\pi \cdot l} \cdot \left(\ln \frac{2 \cdot l}{d} + \frac{1}{2} \ln \frac{4 \cdot t + l}{4 \cdot t - l} \right) \quad (4.4)$$

де ρ - питомий опір ґрунту, залежить від типу грунту, Ом·м;

для піску - А00 ÷ 700 Ом·м; приймаємо ρ = А00 Ом·м;

l - довжина заземлювача, м; для труб - 2-3 м; l = 3 м;

d - діаметр заземлювача, м; для труб - 0,03-0,05 м; d = 0,05 м;

t - відстань від середини забитого в ґрунт заземлювача до рівня землі, м; t = 2 м.

$$R_з = \frac{400}{2 \cdot 3{,}14 \cdot 3} \left(\ln \frac{2 \cdot 3}{0{,}05} + \frac{1}{2} \ln \frac{4 \cdot 2 + 3}{4 \cdot 2 - 3} \right) = 110 \text{, Ом}$$

Опір смуги, що з'єднує заземлювачі, визначається за формулою:

$$R_ш = \frac{\rho}{2\pi \cdot L} \cdot \ln \frac{2 \cdot L^2}{b \cdot t^1} \quad (4.5)$$

де L - довжина смуги, що з'єднує заземлювачі (м) і приблизно дорівнює периметру будівлі: $П_{буд.}$ = 42·2 + 38·2 = 160 м; L = 160 м;

b - ширина смуги, м; b = 0,03 м;

t_1 - глибина заземлення від рівня землі, м; t_1 = 0,5 м.

$$R_п = \frac{400}{2 \cdot 3{,}14 \cdot 160} \cdot \ln \frac{2 \cdot 160^2}{0{,}03 \cdot 0{,}5} = 5{,}99 \text{, Ом}$$

Кількість заземлювачів захисного заземлення визначається за формулою:

$$n = \frac{2 \cdot R_з}{4 \cdot \eta_з} \quad (4.6)$$

де 4 - допустимий загальний опір, Ом;

2 - коефіцієнт сезонності.

Визначаємо загальний опір захисного заземлення:

$$R_{ззп} = \frac{110 \cdot 5{,}99}{5{,}99 \cdot 79 \cdot 0{,}7 + 110 \cdot 0{,}5} = 1{,}7 \text{ Ом}$$

Висновок: дане захисне заземлення буде забезпечувати електробезпеку будівлі, так як виконується умова: $R_{ззп}$ < 4 Ом.

3) При виникненню пожеж при роботі на ПЕОМ від таких можливими джерел запалювання як:

– іскри і дуги коротких замикань;

– перегрів провідників, резисторів та інших радіодеталей ПЕОМ, від тривалої перевантаження та наявність перехідного опору;

– іскри при розмиканні і розмиканні ланцюгів;
– розряди статичної електрики;

необережному поводженню з вогнем, а також вибухи газо-повітряних і паро-повітряних сумішей.

4.7 Охорона навколишнього природного середовища

4.7.1 Загальні дані з охорони навколишнього природного середовища

Діяльність за темою магістерської роботи, а саме: оптимізація запитів до бази даних в процесі її виконання впливає на навколишнє природнє середовище і регламентується нормами діючого законодавства: Законом України «Про охорону навколишнього природного середовища» [8], Законом України «Про забезпечення санітарного та епідемічного благополуччя населення» [9], Законом України «Про відходи» [10].

В процесі роботи виникають процеси поводження з відходами ІТ галузі. Нижче надано перелік відходів, що утворюються в процесі роботи:

- Відпрацьовані люмінесцентні лампи - I клас небезпеки
- Змінні носії інформації - IV клас небезпеки
- Відходи друкуючих пристроїв - IV клас небезпеки
- Макулатура - IV клас небезпеки
- Побутові відходи - IV клас небезпеки

Висновки до четвертого розділу

Проведений аналіз умов праці, шкідливих та небезпечних чинників, з якими стикається робітник.

Визначені параметри і певні характеристики приміщення для роботи над запропонованим в дипломній роботі проектом.

Оописані заходи, які потрібно зробити для того, щоб дане приміщення відповідало необхідним нормам і було комфортним і безпечним для робітника.

Приведені рекомендації щодо організації робочого місця, а також важлива інформація щодо пожежної та електробезпеки.

Наведені розміри приміщення та значення температури, вологості й рухливості повітря, необхідна кількість і потужність ламп та інші параметри, значення яких впливає на умови праці робітника, а також – наведені інструкції з охорони праці, техніки безпеки при роботі на комп’ютері.

Визначені основні екологічні аспекти впливу на навколишнє природнє середовище та зазначені заходи щодо поводження з ними.

Література до четвертого розділу

43. Державні санітарні норми. ДСН 3.3.6.042-99. «Санітарні норми мікроклімату виробничих приміщень. Міністерство охорони здоров'я України (МОЗ)» - Режим доступу: https://zakon.rada.gov.ua/rada/show/va042282-99 - 01.02.1999 р.
44. Державні санітарні правила і норми. ДСанПІН 3.3.2.007-98. «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин. Міністерство охорони здоров'я України (МОЗ)» - Режим доступу: https://zakon.rada.gov.ua/rada/show/v0007282-98 - 10.12.1998 р.
45. Державний стандарт України. ДСТУ Б В.2.5-82:2016 «Електробезпека в будівлях і спорудах. Вимоги до захисних заходів від ураження електричним струмом» - Режим доступу: http://epicentre.co.ua/dstu/doc28522.html - 01.07.2016 р.
46. Міждержавний стандарт. ГОСТ 13109-97. «Норми якості електричної енергії в системах електропостачання загального призначення» - Режим доступу: https://dnaop.com/html/42313/doc-ГОСТ_13109-97 - 21.11.1997 р.
47. НПАОП 0.00-7.15-18. «Вимоги безпеки і захисту здоров'я працівників при роботі з екранними пристроями» - Режим доступу: https://zakon.rada.gov.ua/laws/show/z0508-18 - 14.02.2018 р.
48. Державний стандарт України. ДСТУ Б В.1.1-36:2016. «Визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою» - Режим доступу: https://zakon.rada.gov.ua/rada/show/v0158858-16 - 15.06.2016 р.
49. Міждержавний стандарт. ГОСТ 12.1.044-89. «Система стандартів безпеки праці. Вогнестійкість. Номенклатура показників і методи їх визначення (ІСО 4589-84)» - Режим доступу: http://helpnik.college.ks.ua/standart/gost/Catalog/Index/4/4085.htm - 12.12.1989 р.
50. Закон України «Про охорону навколишнього природного середовища» - Режим доступу: https://zakon.rada.gov.ua/laws/show/1264-12 - 26.06.1991 р.
51. Закони України «Про охорону навколишнього природного середовища» - Режим доступу - https://zakon.rada.gov.ua/laws/show/4004-12 - 24.02.1994 р.
52. Закон України «Про відходи» - Режим доступу: https://zakon.rada.gov.ua/laws/show/187/98-%D0%B2%D1%80 – 05.03.1998 р.

ДОДАТОК А – Презентація

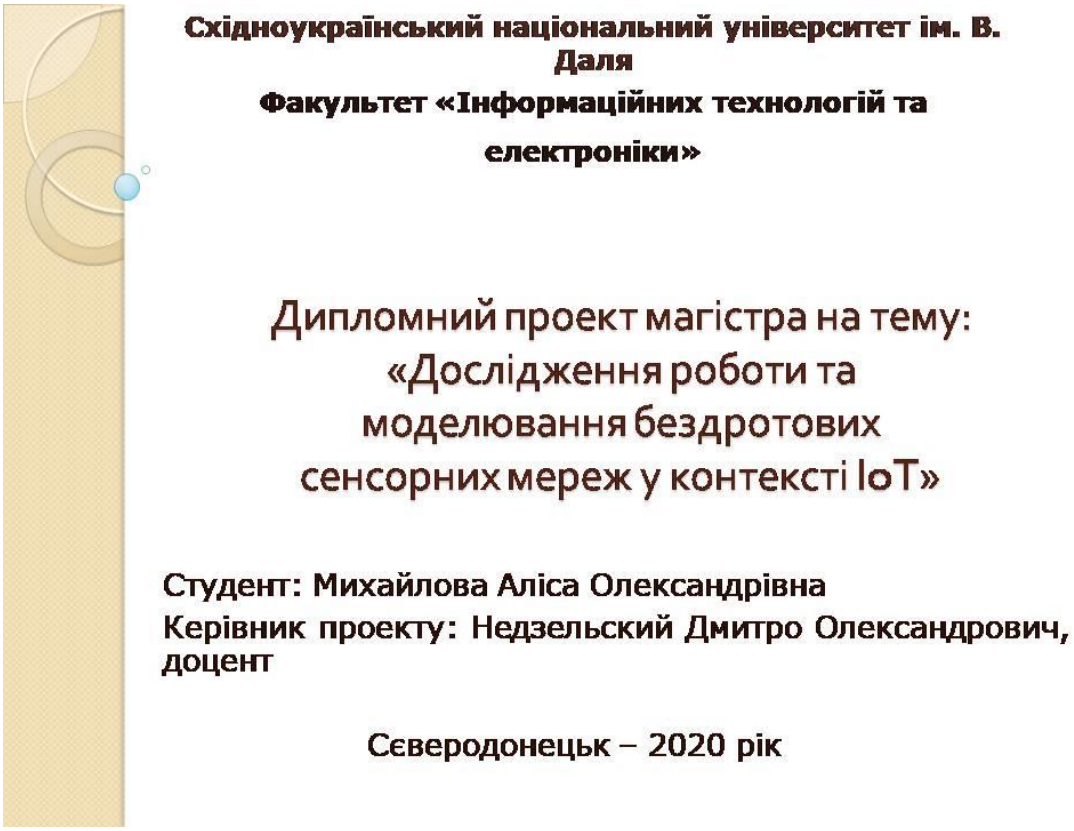


Рисунок А.1 – Титульна сторінка презентації

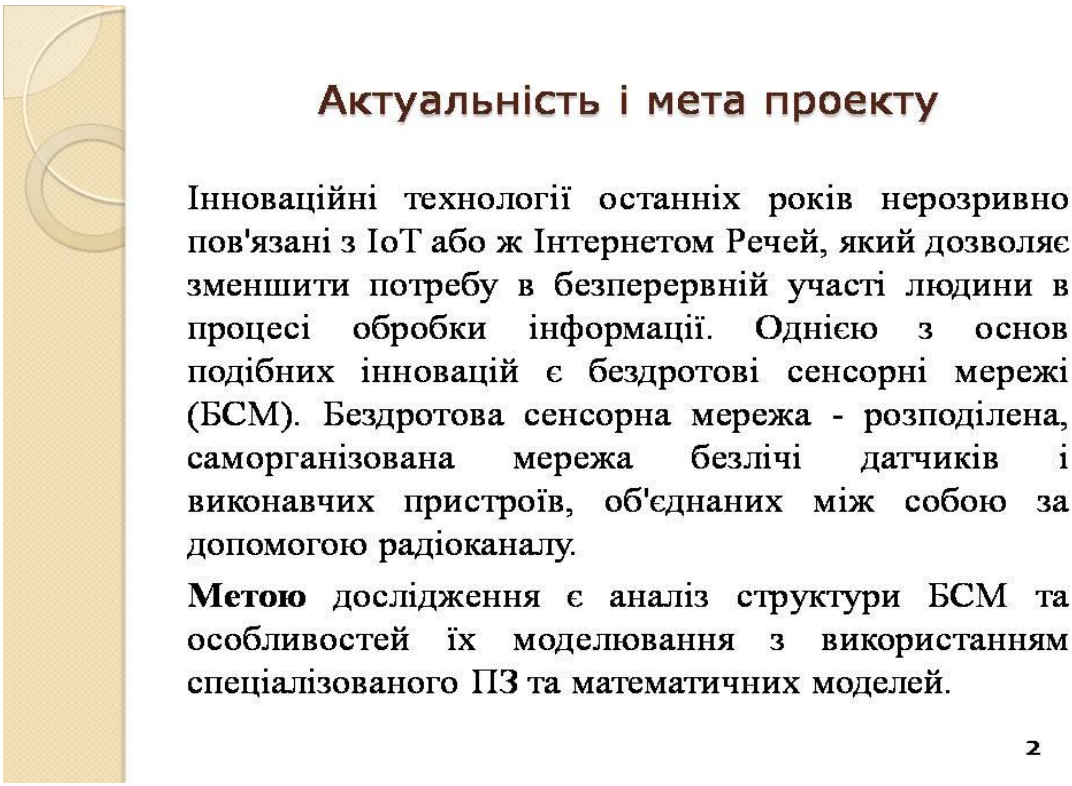


Рисунок А.2 – Сторінка презентації «Актуальність та мета проекту»

Основні питання

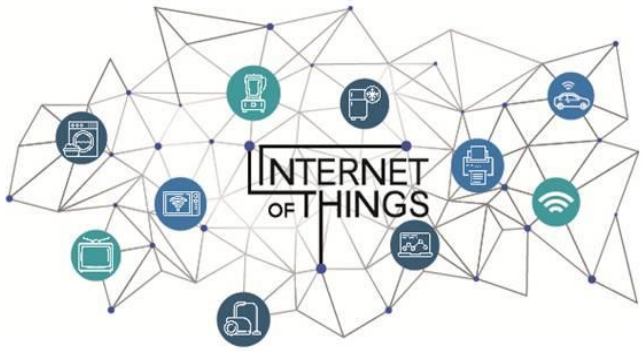
У роботі вирішується питання більш глибокого дослідження БСМ, вибору програмного забезпечення для їх моделювання та перевірки ефективності створеної моделі шляхом аналізу отриманих результатів. Важливою частиною роботи є порівняльний аналіз протоколів маршрутизації, які були використані при моделюванні.

3

Рисунок А.3 – Сторінка презентації «Основні питання»

БСМ та IoT

Інтернет – речей (IoT), як технологія, своїй появі зобов'язана імплементації технології радіочастотного обміну між пристроями, здатними комунікувати в загальній мережі. Вся концепція IoT будувалася на використанні технології радіочастотної ідентифікації і БСМ.



4

Рисунок А.4 – Сторінка презентації «БСМ та IoT»

Рисунок А.5 – Сторінка презентації «Сфери використання БСМ»

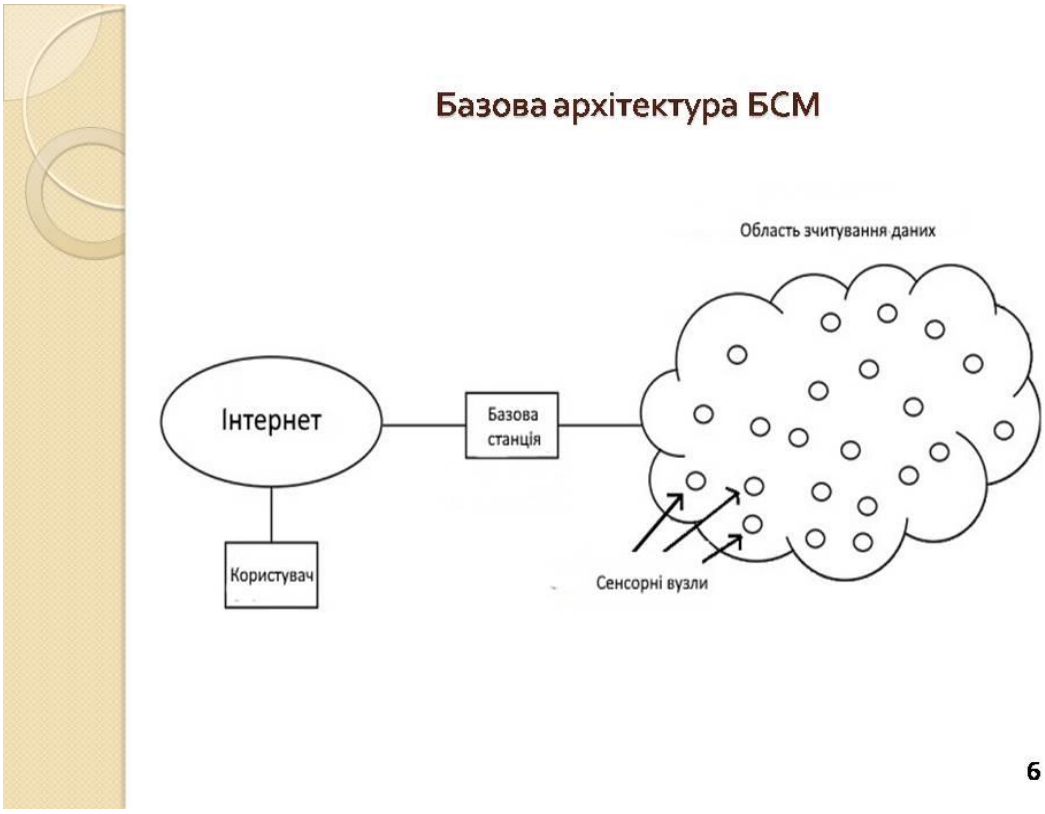


Рисунок А.6 – Сторінка презентації «Базова архітектура БСМ»

Рисунок А.7 – Сторінка презентації «Стандарт БСМ та його специфікації»

Категорія протоколів	Протоколи
Засновані на місцезнаходженні вузлів	MECN, SMECN, GAF, GEAR, Span, TBF, BVGF, GeRaF
Направлені на агрегацію даних	SPIN, Directed Diffusion, Rumor Routing, COUGAR, ACQUIRE, EAD
Ієрархічні	LEACH, PEGASIS, HEED, TEEN, APTEEN
Засновані на мобільності	SEAD, TTDD, Joint Mobility and Routing, Data Mules, Dynamic Proxy Tree-Base Data Dissemination
Мульти-орієнтовані	Sensor-Disjoint Multipath, Braided Multipath, N-to-1 Multipath Discovery
Засновані на гетерогенності	IDSQ, CADR, CHR
Засновані на якості обслуговування	SAR, SPEED, Energy-aware routing

Ознака	LEACH	TEEN	HEED
Енергоефективність	Дуже низька	Дуже висока	Середня
Затримка	Дуже невелика	Невелика	Середня
Стабільність кластера	Середня	Висока	Висока
Масштабованість	Дуже низька	Низька	Середня
Врівноваження навантаження	Середня	Висока	Середня
Складність алгоритму	Низька	Висока	Середня

Рисунок А.8 – Сторінка презентації «Протоколи БСМ»

Існуючі програмні засоби для моделювання БСМ

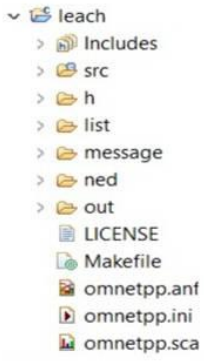
- NS-2
- TOSSIM
- OMNeT++

Назва системи	Графічний інтерфейс	Орієнтованість спеціалізації на БСМ	Можливість побудови БСМ будь-якої складності
NS-2	Ні	Ні	Ні
TOSSIM	Так	Так	Ні
OMNeT++	Так	Так	Так

9

Рисунок А.9 – Сторінка презентації «Існуючі засоби для моделювання БСМ»

Принципи роботи OMNeT++



10

Рисунок А.10 – Сторінка презентації «Принципи роботи OMNeT++»

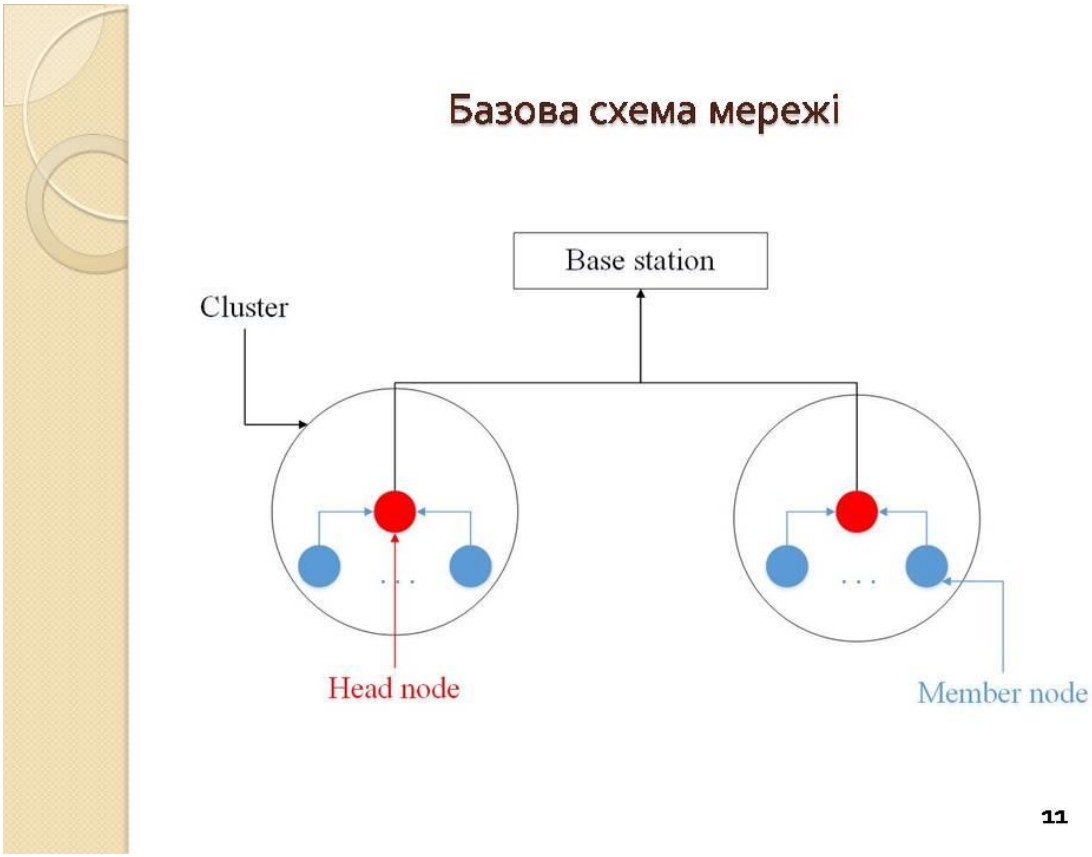


Рисунок А.11 – Сторінка презентації «Базова схема мережі»

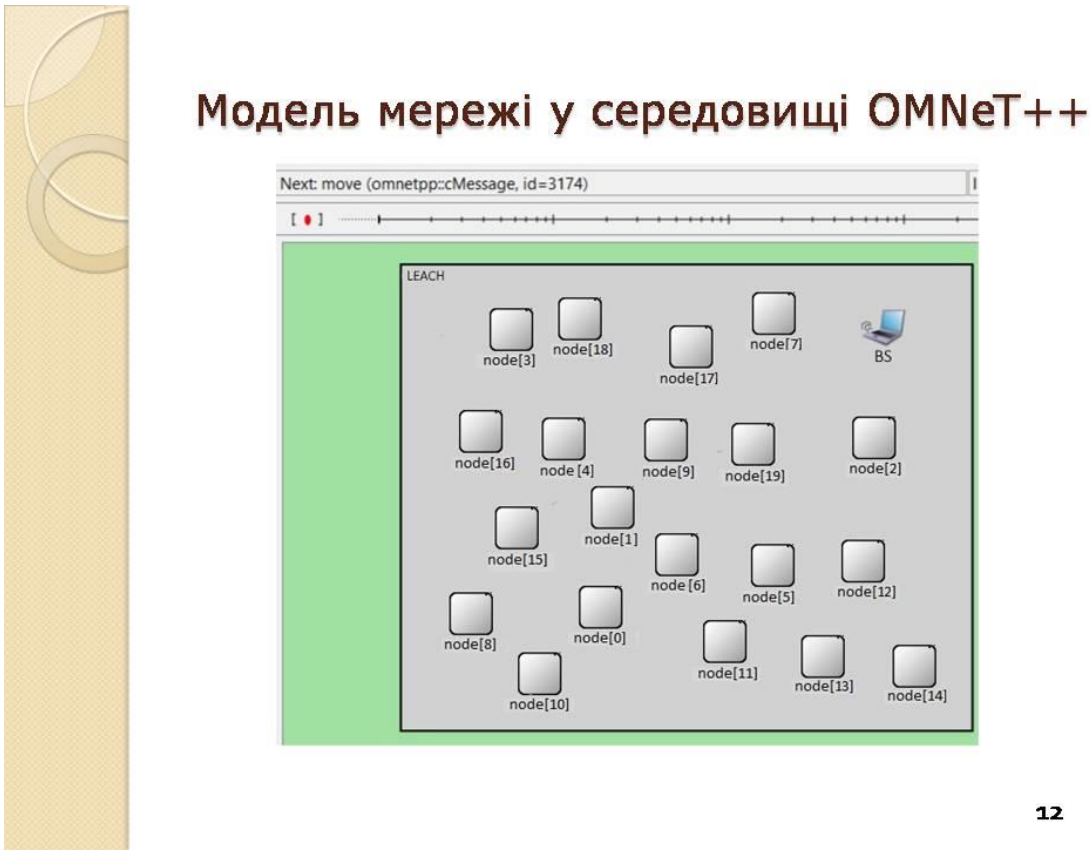


Рисунок А.12 – Сторінка презентації «Модель мережі у середовищі OMNeT++»

Рисунок А.13 – Сторінка презентації «Схема налаштування LEACH»

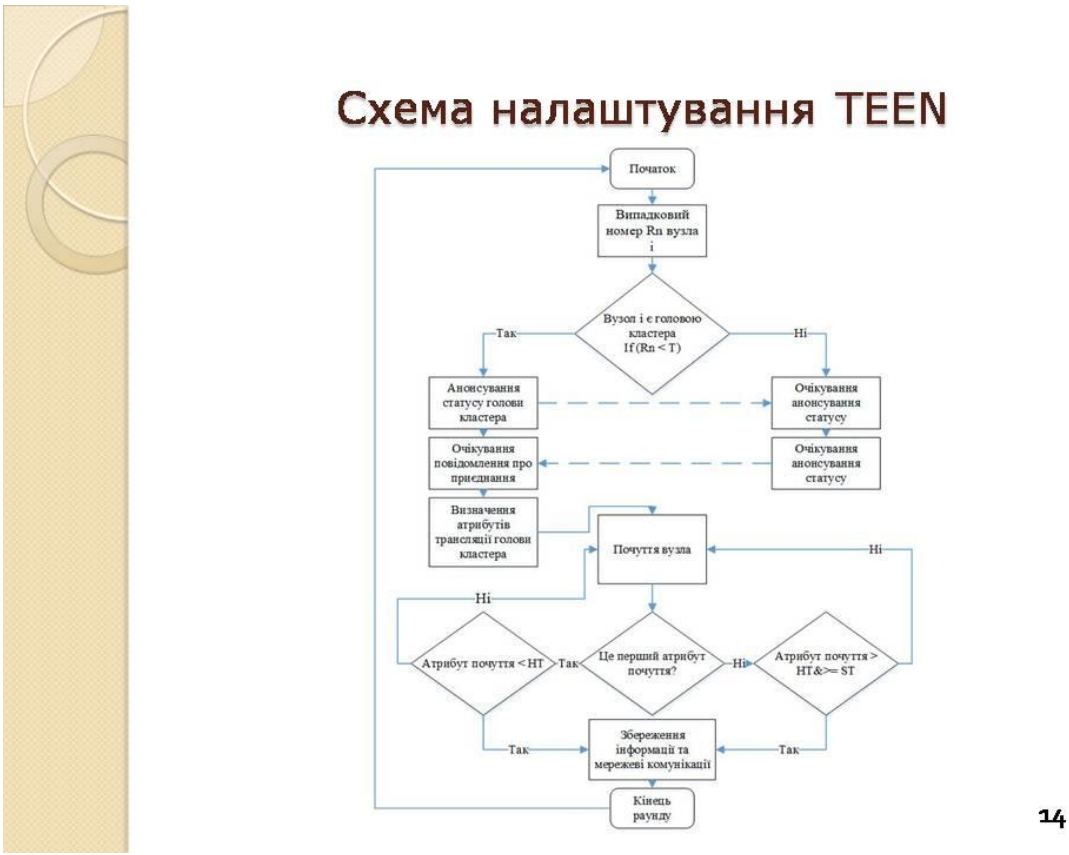


Рисунок А.14 – Сторінка презентації «Схема налаштування TEEN»

Рисунок А.15 – «Схема налаштування HEED»

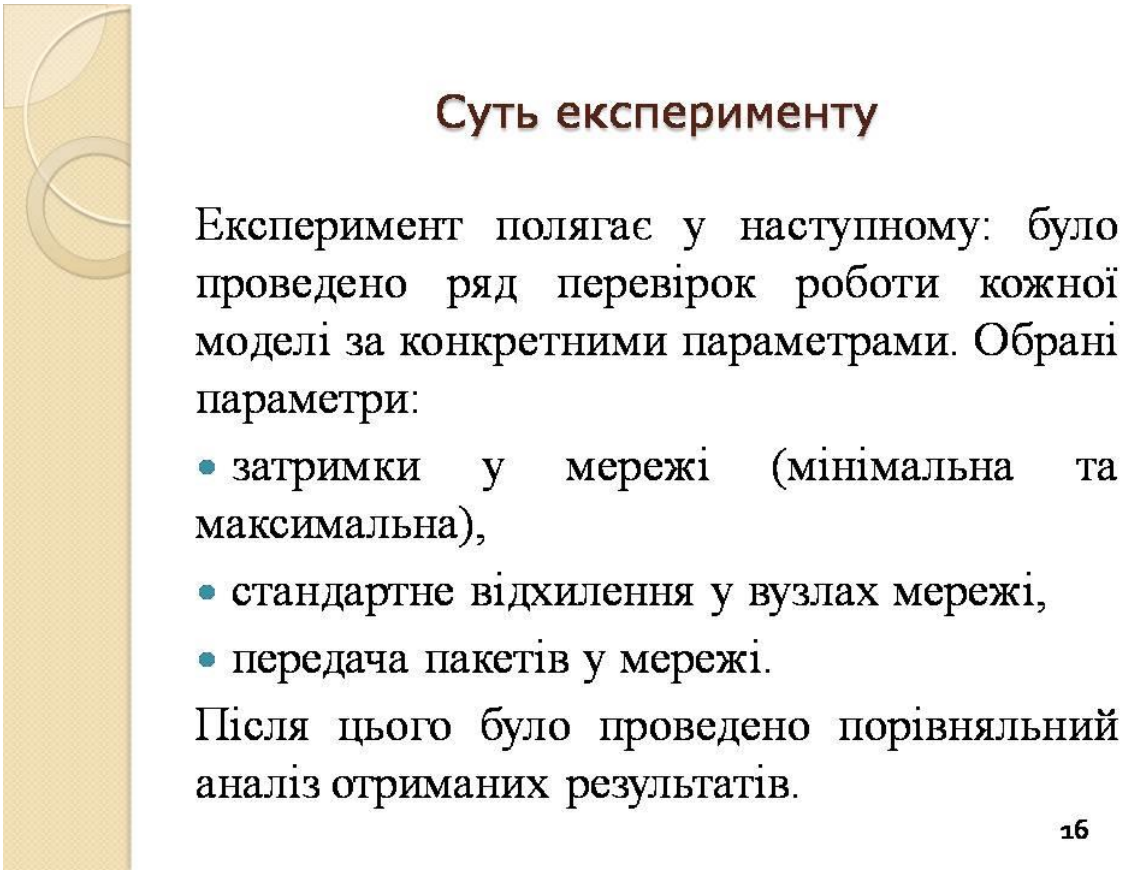


Рисунок А.16 – Сторінка презентації «Суть екперименту»

Рисунок А.17 – Сторінка презентації «Затримки (максимальна та мінімальна)»

Рисунок А.18 – Сторінка презентації «Стандартне вхиленя та передача пакетів у мережі»

Математичний аналіз

Параметром аналізу були обрані затримки у вузлах мережі та формула критерію сумарної середньої затримки

LEACH $\sum_{i=1}^{20}(3.36 + 0.49) = 77$

TEEN $\sum_{i=1}^{20}(2.135 + 0.325) = 49{,}2$

HEED $\sum_{i=1}^{20}(1.83 + 0.32) = 43$

19

Рисунок А.19 – Сторінка презентації «Математичний аналіз»

Висновки

У результаті виконання дипломної роботи було проаналізовано особливості роботи та моделювання БСМ. Практично перевірена функціональність середовища моделювання OMNeT++, проаналізована його внутрішня структура та особливості побудови проектів. Для цього було створено модель бездротової сенсорної мережі, яка була налаштована за допомогою заздалегідь обраних протоколів маршрутизації LEACH, TEEN та HEED, після чого було проведено ряд перевірок працездатності отриманих моделей.

Моделі були перевірені за наступними критеріями:

- **затримки у вузлах мережі;**
- **стандартне відхилення у вузлах мережі;**
- **передача пакетів у вузлах мережі.**

Моделі перевірені за допомогою математичних моделей.

Обраний найбільш вигідний з точки зору практичного застосування протокол маршрутизації – HEED.

Сформульовані висновки щодо середовища моделювання OMNeT++, згідно з якими середовище є повністю відповідним до вимог, які було висунуто до подібного ПЗ.

20

Рисунок А.20 – Сторінка презентації «Висновки»

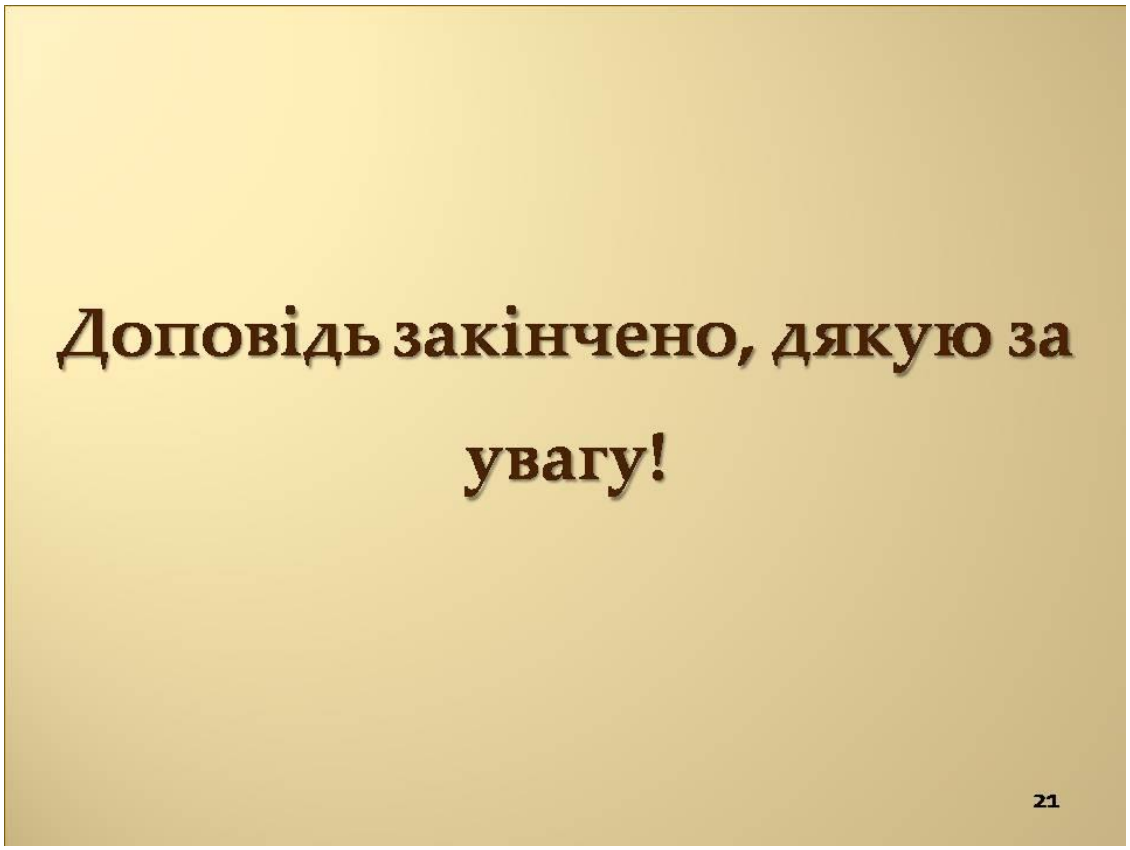


Рисунок А.21 – Сторінка презентації «Висновки»

ДОДАТОК Б – Вихідний код

Файл leach.cpp

```
1. #include <fstream>
2. #include <stdlib.h>
3. #include <time.h>
4. #include "Leach.h"
5. using namespace std;
6. Leach::Leach(double mP, unsigned int N, unsigned int Time)
7. :P(mP), NodeCount(N), TimeLapse(Time)
8. {
9. this->NodeList = new Node[N];
10. for (unsigned int i = 0; i < NodeCount; i++)
11. NodeList[i].SetNode(NodeCount, i);
12. }
13. Leach::~Leach(void)
14. {
15. if (NodeList) delete []NodeList;
16. }
17. void Leach::LoadNodes()
18. {
19. fstream fs;
20. fs.open("input.txt");
21. if (fs.is_open())
22. {
23. for (unsigned int i = 0; i < NodeCount; i++)
24. {
25. fs >> NodeList[i].px >> NodeList[i].py;
26. }
27. fs.close();
28. }
29. }
30. void Leach::StartWork()
31. {
32. srand((unsigned)time( NULL ));
```

```
33. for (unsigned int i = 0; i < NodeCount; i++)
34. {
35. NodeList[i].SetDistance(NodeList);
36. NodeList[i].SetE(4.00);
37. cout<<i<<"?????:"<<NodeList[i].ne<<endl;
38. }
39. int Remain = NodeCount;
40. double T;
41. int r = 0;
42. double s;
43. while(Remain)
44. {
45. r ++;
46. cout <<"----------No." << r << "----------"<<endl;
47. if (Remain > 1)
48. {
49. T = P / (1 - r % (int)(1/P) * P );
50. for (unsigned int i = 0; i < NodeCount; i++)
51. {
52. if (!NodeList[i].IsCluster)
53. {
54. s = (double)rand() / RAND_MAX;
55. if (s < T)
56. {
57. NodeList[i].IsCluster = true;
58. NodeList[i].IsCurrentCluster = true;
59. NodeList[i].OnBecomeCluster();
60. Remain --;
61. }
62. else
63. {
64. NodeList[i].IsCurrentCluster = false;
65. }
66. }
67. else{
```

```
68. NodeList[i].IsCurrentCluster = false;
69. }
70. }
71. }
72. else
73. {
74. for (unsigned int i = 0; i < NodeCount; i++)
75. {
76. if (!NodeList[i].IsCluster){
77. NodeList[i].IsCluster = true;
78. NodeList[i].IsCurrentCluster = true;
79. NodeList[i].OnBecomeCluster();
80. Remain --;
81. }
82. else{
83. NodeList[i].IsCurrentCluster = false;
84. }
85. }
86. }
87. cout <<endl;
88. for (unsigned int i = 0; i < NodeCount && Remain; i++)
89. {
90. if (!NodeList[i].IsCurrentCluster)
91. {
92. int clusterHead = NodeList[i].OnNoneCluster();
93. cout<<"??"<<i<<"???"<<clusterHead<<"?????"<<endl
94. Node r;
95. int u = r.consumeE(NodeList[i],NodeList[clusterHead]);
96. cout<<NodeList[i].GetE()<<" "<<NodeList[clusterHead].GetE()<<endl;
97. }
98. }
99. }
100.          }
```

Файл Teen.cpp

```
1. #include <teen/teen_packet.h>
2. #include <random.h>
3. #include <cmu-trace.h>
4. #include <math.h>
5. #include <unistd.h>
6. #define DEBUG
7. int hdr_teen::offset_;
8. static class TEENHeaderClass : public PacketHeaderClass
9. {
10. public:
11. TEENHeaderClass() : PacketHeaderClass("PacketHeader/TEEN", sizeof(hdr_teen))
12. {
13. bind_offset(&hdr_teen::offset_);
14. }
15. } class_rtProtoTEEN_hdr;
16. static class TEENclass : public TclClass
17. {
18. public:
19. TEENclass() : TclClass("Agent/TEEN") {}
20. TclObject* create(int argc, const char*const* argv)
21. {
22. assert(argc == 5);
23. return (new TEEN((nsaddr_t)Address::instance().str2addr(argv[4])));
24. }
25. } class_rtProtoTEEN;
26. TEEN::TEEN(nsaddr_t id) : Agent(PT_TEEN)
27. {
28. #ifdef DEBUG
29. printf("N node is set to %d, start broadcasting  \n",index);
30. #endif
31. index=id;
32. rlink_=-1;
33. flink_=-1;
```

```
34. flag_=0;
35. ultimated_=0;
36. result_=0;
37. seqno=1;
38. }
39. int TEEN::command(int argc, const char*const* argv)
40. {
41. if(argc==4)
42. {
43. if(strncasecmp(argv[1], "start", 5) == 0)
44. {
45. nsaddr_t source = atoi(argv[2]);
46. nsaddr_t dest = atoi(argv[3]);
47. printf("Source = %d",source);
48. printf("Dest = %d\n",dest);
49. if(dest != 0)
50. send_req(source,dest);
51. return TCL_OK;
52. #ifdef DEBUG
53. printf("Node is set to %d, start broadcasting  \n", index);
54. #endif
55. }
56. }
57. if (argc == 2 )
58. {
59. if (strcasecmp(argv[1],"start") == 0)
60. {
61. nsaddr_t  source=atoi(argv[2]);
62. printf("source=%d",source);
63. call_creset(source);
64. return TCL_OK;
65. }
66. }
67. if (argc == 3)
68. {
```

```
69. if (strcmp(argv[1], "index") == 0)
70. {
71. index = atoi(argv[2]);
72. return TCL_OK;
73. }
74. else if (strncasecmp(argv[1], "start", 5) == 0)
75. {
76. nsaddr_t  source=atoi(argv[2]);
77. printf("source=%d",source);
78. call_creset(source);
79. return TCL_OK;
80. #ifdef DEBUG
81. printf("N : node is set to %d, start broadcasting  \n", index);
82. #endif
83. }
84. else if(strcmp(argv[1], "log-target") == 0 || strcmp(argv[1], "tracetarget") == 0)
85. {
86. logtarget = (Trace*) TclObject::lookup(argv[2]);
87. if(logtarget == 0)
88. return TCL_ERROR;
89. return TCL_OK;
90. }
91. else if(strcmp(argv[1], "drop-target") == 0)
92. {
93. return Agent::command(argc, argv);
94. }
95. else if(strcmp(argv[1], "if-queue") == 0)
96. {
97. ifqueue = (PriQueue*) TclObject::lookup(argv[2]);
98. if(ifqueue == 0)
99. return TCL_ERROR;
100.        return TCL_OK;
101.        }
102.        else if (strcmp(argv[1], "port-dmux") == 0)
103.        {
```

```
104.    dmux_ = (PortClassifier *)TclObject::lookup(argv[2]);
105.    if (dmux_ == 0)
106.    {
107.    fprintf (stderr, "%s: %s lookup of %s failed\n", __FILE__,argv[1], argv[2]);
108.    return TCL_ERROR;
109.    }
110.    return TCL_OK;
111.    }
112.    }
113.    return Agent::command(argc, argv);
114.    }
115.    void TEEN :: send_req(nsaddr_t src, nsaddr_t dest)
116.    {
117.    printf("\n\nSource=%d\n",src);
118.    if(rlink_==-1)
119.    {
120.    if(flag_==0)
121.    {
122.    flag_=1;
123.    }
124.    }
125.    ultimated_=dest;
126.    if(index==src)
127.    {
128.    Packet *p = Packet::alloc();
129.    struct hdr_cmn *ch = HDR_CMN(p);
130.    struct hdr_ip *ih = HDR_IP(p);
131.    struct hdr_teen_req *req = HDR_TEEN_REQ(p);
132.    #ifdef DEBUG
133.    printf("sending request from %d\n", index);
134.    #endif

135.    // Write Channel Header
136.    ch->ptype() = PT_TEEN;
137.    ch->direction()=hdr_cmn::DOWN;
```

```
138.        ch->next_hop()=IP_BROADCAST;
139.        ch->size() = IP_HDR_LEN + req->size();
140.        //ch->next_hop() = IP_BROADCAST;
141.        ch->addr_type() = NS_AF_NONE;
142.        ch->prev_hop_ = index;
143.        // Write IP Header
144.        ih->saddr() = index;
145.        ih->daddr() = IP_BROADCAST;
146.        ih->sport() = RT_PORT;
147.        ih->dport() = RT_PORT;
148.        ih->ttl_ = NETWORK_DIAMETER;
149.        // Write req Header
150.        req->pkt_type = TEEN_REQ;
151.        req->src_nodeid = index;
152.        req->pkt_id = seqno;
153.        seqno += 1;
154.        Scheduler::instance().schedule(target_,p,JITTER);
155.        #ifdef DEBUG
156.        #endif
157.        }
158.        }
159.        void TEEN::recv_req(Packet *p)
160.        {
161.        struct hdr_teen_req *req = HDR_TEEN_REQ(p);
162.        // I have originated the packet, just drop it
163.        if(req->src_nodeid== index)
164.        { //if source is myself
165.        Packet::free(p);
166.        return;
167.        }
168.        if(flag_ == 0)
169.        {
170.        flag_=1;
171.        rlink_=req->src_nodeid;
172.        printf("Rlink of %d is %d\n",index,rlink_);
```

```
173.        send_resp(rlink_,index);
174.        }
175.        else
176.        {
177.        printf("\nNode %d has already a leader %d\n",index,rlink_);
178.        Packet::free(p);
179.        return;
180.        }
181.        }
182.        void TEEN::send_resp(nsaddr_t tnodeid_,nsaddr_t pnodeid_)
183.        {
184.        Packet *p = Packet::alloc();
185.        struct hdr_cmn *ch = HDR_CMN(p);
186.        struct hdr_ip *ih = HDR_IP(p);
187.        struct hdr_teen_resp *resp = HDR_TEEN_RESP(p);
188.        ch->ptype() = PT_TEEN;
189.        ch->direction()=hdr_cmn::DOWN;
190.        ch->next_hop()=tnodeid_;
191.        ch->size() = IP_HDR_LEN + resp->size();
192.        ch->addr_type() = NS_AF_INET;
193.        ch->prev_hop_ = pnodeid_;
194.        // Write IP Header
195.        ih->saddr() = index;
196.        ih->daddr() = rlink_;
197.        ih->sport() = RT_PORT;
198.        ih->dport() = RT_PORT;
199.        ih->ttl_ = NETWORK_DIAMETER;
200.        // Write resp Header
201.        resp->pkt_type = TEEN_RESP;
202.        resp->desti_nodeid = rlink_;
203.        resp->src_nodeid =index;
204.        resp->pkt_id = seqno;
205.        seqno += 1;
206.        Scheduler::instance().schedule(target_, p, JITTER);
207.        #ifdef DEBUG
```

```
208.        printf("send response by %d to %d \n",resp->src_nodeid,resp->desti_nodeid);
209.        #endif
210.        }
211.        void TEEN::recv_resp(Packet *p)
212.        {
213.        struct hdr_teen_resp *resp = HDR_TEEN_RESP(p);
214.        if(resp->src_nodeid == index)
215.        { //if source is myself
216.        Packet::free(p);
217.        return;
218.        }
219.        {
220.        printf("\nReceived response by %d from %d \n",resp->desti_nodeid,resp->src_nodeid);
221.        if(resp->src_nodeid == ultimated_)
222.        {
223.        if(rlink_==-1)
224.        {
225.        flink_=resp->src_nodeid;
226.        printf("Route connected");
227.        }
228.        else
229.        {
230.        result_=1;
231.        flink_=resp->src_nodeid;
232.        printf("Flink of %d is %d",index,flink_);
233.        send_result(rlink_,result_);
234.        }
235.        }
236.        else if(resp->src_nodeid != ultimated_)
237.        {
238.        send_cont(resp->src_nodeid,ultimated_);
239.        }
240.        }
241.        }
242.        void TEEN::send_cont(nsaddr_t desti_,nsaddr_t ulti_)
```

```
243.        {
244.        Packet *p = Packet::alloc();
245.        struct hdr_cmn *ch = HDR_CMN(p);
246.        struct hdr_ip *ih = HDR_IP(p);
247.        struct hdr_teen_cont *cont = HDR_TEEN_CONT(p);
248.        // Write Channel Header
249.        ch->ptype() = PT_TEEN;
250.        ch->direction()=hdr_cmn::DOWN;
251.        ch->next_hop()=desti_;
252.        ch->size() = IP_HDR_LEN + cont->size();
253.        ch->addr_type() = NS_AF_INET;
254.        ch->prev_hop_ = index;
255.        // Write IP Header
256.        ih->saddr() = index;
257.        ih->daddr() = desti_;
258.        ih->sport() = RT_PORT;
259.        ih->dport() = RT_PORT;
260.        ih->ttl_ = NETWORK_DIAMETER;
261.        // Write Confirm Header
262.        cont->pkt_type = TEEN_CONT;
263.        //cont->desti_nodeid = desti_;
264.        cont->src_nodeid= index;
265.        //want to write ultimate destination here
266.        cont->udesti_nodeid =ulti_ ;
267.        cont->pkt_id = seqno;
268.        seqno += 1;
269.        Scheduler::instance().schedule(target_, p, 0.0);
270.        #ifdef DEBUG
271.        printf(" \ncont pkt sent by %d to %d  \n",index,desti_);
272.        #endif
273.        }
274.        void TEEN::recv_cont(Packet *p)
275.        {
276.        struct hdr_teen_cont *cont = HDR_TEEN_CONT(p);
277.        ultimated_=cont->udesti_nodeid;
```

```
278.        if (cont->src_nodeid == rlink_)
279.        {
280.        printf("Recieved cont packet from %d\n",cont->src_nodeid);
281.        send_req(index,ultimated_);
282.        }
283.        else
284.        {
285.        Packet::free(p);
286.        return;
287.        }
288.        }
289.        void TEEN::send_result(nsaddr_t leader_,u_int8_t res)
290.        {
291.        Packet *p = Packet::alloc();
292.        struct hdr_cmn *ch = HDR_CMN(p);
293.        struct hdr_ip *ih = HDR_IP(p);
294.        struct hdr_teen_result *result = HDR_TEEN_RESULT(p);
295.        //channel header
296.        ch->ptype() = PT_TEEN;
297.        ch->direction()=hdr_cmn::DOWN;
298.        ch->next_hop()=leader_;
299.        ch->size() = IP_HDR_LEN + result->size();
300.        ch->addr_type() = NS_AF_INET;
301.        ch->prev_hop_ = index;
302.        // Write IP Header
303.        ih->saddr() = index;
304.        ih->daddr() = leader_;
305.        ih->sport() = RT_PORT;
306.        ih->dport() = RT_PORT;
307.        ih->ttl_ = NETWORK_DIAMETER;
308.        //write result header
309.        result->pkt_type=TEEN_RESULT;
310.        result->pkt_id=seqno;
311.        result->src_nodeid=index;
312.        //write the status of the route
```

```
313.        result->route_info=res;
314.        seqno+=1;
315.        Scheduler::instance().schedule(target_, p, 0.0);
316.        #ifdef DEBUG
317.        printf(" sent result pkt by %d  to %d \n",index,leader_);
318.        #endif
319.        }
320.        void TEEN::call_creset(nsaddr_t src)
321.        {
322.        if(index == src)
323.        {
324.        Packet *p = Packet::alloc();
325.        struct hdr_cmn *ch = HDR_CMN(p);
326.        struct hdr_ip *ih = HDR_IP(p);
327.        struct hdr_teen_creset *rt = HDR_TEEN_CRESET(p);
328.        printf("Sending RESET from %d \n", index);

329.        // Write Channel Header
330.        ch->ptype() = PT_TEEN;
331.        ch->direction()=hdr_cmn::DOWN;
332.        ch->next_hop()=IP_BROADCAST;
333.        ch->size() = IP_HDR_LEN + rt->size();
334.        //ch->next_hop() = IP_BROADCAST;
335.        ch->addr_type() = NS_AF_NONE;
336.        ch->prev_hop_ = index;
337.        // Write IP Header
338.        ih->saddr() = index;
339.        ih->daddr() = IP_BROADCAST;
340.        ih->sport() = RT_PORT;
341.        ih->dport() = RT_PORT;
342.        ih->ttl_ = NETWORK_DIAMETER;
343.        // Write breq Header
344.        rt->pkt_type = TEEN_CRESET;
345.        rt->src_nodeid = index;
346.        rt->pkt_id = seqno;
```

```
347.        seqno += 1;
348.        Scheduler::instance().schedule(target_,p,JITTER);
349.        }
350.        }
351.        void TEEN::recv_creset(Packet *p)
352.        {
353.        struct hdr_teen_creset *rt = HDR_TEEN_CRESET(p);
354.        if (rt->src_nodeid== index)
355.        {
356.        Packet::free(p);
357.        return;
358.        }
359.        if (flag_== 0||rlink_==-1)
360.        {
361.        Packet::free(p);
362.        return;
363.        }
364.        else if(flag_!=0||rlink_!=-1)
365.        {
366.        printf("Myid=%d\n",index);
367.        rlink_=-1;
368.        flink_=-1;
369.        flag_=0;
370.        ultimated_=0;
371.        result_=0;
372.        seqno=1;
373.        call_creset(index);
374.        }
```

Файл Heed.cpp

```
1. include Heed;
2. include HeedCmd;
3. #include <math.h>
4. #include <unistd.h>
5. module HeedM {
```

```
6.  provides {
7.  interface StdControl;
8.  }
9.  uses {
10. interface ADC;
11. interface Timer;
12. interface Leds;
13. interface StdControl as Sounder;
14. interface Send;
15. interface Receive as Bcast;
16. interface RouteControl;
17. interface EnergyControl;
18. interface Intercept as InterceptHeedMsg;
19. }
20. }
21. implementation {
22. enum {
23. TIMER_GETADC_COUNT = 1,// Timer ticks for ADC
24. TIMER_CHIRP_COUNT = 10,// Timer on/off chirp count
25. };
26. bool sleeping;                    // application command state
27. bool focused;
28. bool rebroadcast_adc_packet;
29. TOS_Msg gMsgBuffer;
30. norace uint16_t gSensorData;      // protected by gfSendBusy flag
31. bool gfSendBusy;
32. int timer_rate;
33. int timer_ticks;
34. #define COLLECT_COUNT 1
35. uint16_t numCollectedPoints, numSentPoints, motesAlive, sendMotesAlive;
36. int CH_timer_ticks, reportNumber;
37. static void initialize() {
38. timer_rate = INITIAL_TIMER_RATE;
39. atomic gfSendBusy = FALSE;
40. sleeping = FALSE;
```

```
41. rebroadcast_adc_packet = FALSE;
42. focused = FALSE;
43. /*** : clustering and control ******/
44. numCollectedPoints = numSentPoints = 0;
45. motesAlive = sendMotesAlive = 0;
46. CH_timer_ticks = 0;
47. reportNumber = 0;
48. }
49. task void SendData() {
50. HeedMsg *pReading;
51. uint16_t Len;
52. dbg(DBG_USR1, "HeedM: Sending sensor reading\n");
53. if (pReading = (HeedMsg *)call Send.getBuffer(&gMsgBuffer,&Len)) {
54. pReading->type = HEED_TYPE_SENSORREADING;
55. pReading->parentaddr = call RouteControl.getParent();
56. //pReading->reading = gSensorData;
57.  pReading->reading = reportNumber;                // use gSensorData for real data collection
58. /*****code **********/
59. pReading->remPower = call EnergyControl.getRemainingPoints();   // remaining power
60. pReading->overhead = call EnergyControl.getOverhead();          // overhead so far
61. pReading->nPoints = numSentPoints;
62. pReading->motesAlive = sendMotesAlive;
63. /*************************/
64. if ((call Send.send(&gMsgBuffer,sizeof(HeedMsg))) != SUCCESS)
65. atomic gfSendBusy = FALSE;
66. }
67. }
68. command result_t StdControl.init() {
69. initialize();
70. return SUCCESS;
71. }
72. command result_t StdControl.start() {
73. return call Timer.start(TIMER_REPEAT, timer_rate);
74. return SUCCESS;
75. }
```

```
76. command result_t StdControl.stop() {
77. return call Timer.stop();
78. }
79. event result_t Timer.fired() {
80. // make sure there is still some power left
81. if (!(call EnergyControl.isAlive())) {
82. call StdControl.stop();
83. return SUCCESS;
84. }
85. #ifdef CLUSTERING_ON
86. // Case 1. Clustering is being applied ...
87. // generate your locally sensed data
88. if (TOS_LOCAL_ADDRESS != 0) {
89. numCollectedPoints++;
90. motesAlive |= (1 << (TOS_LOCAL_ADDRESS-1));
91. }
92. CH_timer_ticks++;
93. // If no parent is available, continue holding the data.
94. if (call RouteControl.getParent() == 0xFFFF) {
95. return SUCCESS;
96. }
97. // If cluster head, send aggregated data
98. if (call RouteControl.isClusterHead()) {
99. if (CH_timer_ticks % COLLECT_COUNT == 0) {
100.  atomic {
101.  numSentPoints = numCollectedPoints;
102.  sendMotesAlive = motesAlive;
103.  motesAlive = 0;
104.  if (TOS_LOCAL_ADDRESS != 0)
105.  numCollectedPoints = 0;
106.  reportNumber++;
107.  if (!gfSendBusy) {
108.  gfSendBusy = TRUE;
109.  post SendData();
110.  }
```

```
111.  }
112.  }
113.  }
114.  // If not cluster head, send your reading
115.  else {
116.  atomic {
117.  numSentPoints = numCollectedPoints;
118.  sendMotesAlive = motesAlive;
119.  motesAlive = 0;
120.  if (TOS_LOCAL_ADDRESS != 0)
121.  numCollectedPoints = 0;
122.  reportNumber++;
123.  if (!gfSendBusy) {
124.  gfSendBusy = TRUE;
125.  post SendData();
126.  }
127.  }
128.  }
129.  #else
130.  // Case 2. Clustering is not applied
131.  // Base station send aggregated data
132.  // Regular motes simply forward their readings.
133.  atomic {
134.  if (TOS_LOCAL_ADDRESS == 0){
135.  //numSentPoints++;
136.  atomic {
137.  numSentPoints += numCollectedPoints;
138.  numCollectedPoints = 0;
139.  }
140.  }
141.  else
142.  numSentPoints = 1;
143.  reportNumber++;
144.  if (!gfSendBusy) {
145.  gfSendBusy = TRUE;
```

```
post SendData();
}
}
call Leds.redToggle();
#endif
/*dbg(DBG_USR1, "HeedM: Timer fired\n");
timer_ticks++;
if (timer_ticks % TIMER_GETADC_COUNT == 0) {
call ADC.getData();
}
// If we're the focused node, chirp
if (focused && timer_ticks % TIMER_CHIRP_COUNT == 0) {
call Sounder.start();
}
// If we're the focused node, chirp
if (focused && timer_ticks % TIMER_CHIRP_COUNT == 1) {
call Sounder.stop();
}*/
return SUCCESS;
}
async event result_t ADC.dataReady(uint16_t data) {
//HeedMsg *pReading;
//uint16_t Len;
dbg(DBG_USR1, "HeedM: Got ADC reading: 0x%x\n", data);
atomic {
if (!gfSendBusy) {
    a. gfSendBusy = TRUE;
    b. gSensorData = data;
    c. post SendData();
}
}
return SUCCESS;
}
event result_t Send.sendDone(TOS_MsgPtr pMsg, result_t success) {
dbg(DBG_USR2, "HeedM: output complete 0x%x\n", success);
```

```
178. //call Leds.greenToggle();
179. atomic gfSendBusy = FALSE;
180. return SUCCESS;
181. }
182. event TOS_MsgPtr Bcast.receive(TOS_MsgPtr pMsg, void* payload, uint16_t payloadLen){
183. HeedCmdMsg *pCmdMsg = (HeedCmdMsg *)payload;
184. dbg(DBG_USR2, "HeedM: Bcasttype 0x%02x\n", pCmdMsg->type);
185. if (pCmdMsg->type == HEED_TYPE_SETRATE) { // Set timer rate
186. timer_rate = pCmdMsg->args.newrate;
187. dbg(DBG_USR2, "HeedM: set rate %d\n", timer_rate);
188. call Timer.stop();
189. call Timer.start(TIMER_REPEAT, timer_rate);
190. } else if (pCmdMsg->type == HEED_TYPE_SLEEP) {
191. // Go to sleep - ignore everything until a HEED_TYPE_WAKEUP
192. dbg(DBG_USR2, "HeedM: sleep\n");
193. sleeping = TRUE;
194. call Timer.stop();
195. call Leds.greenOff();
196. call Leds.yellowOff();
197. } else if (pCmdMsg->type == HEED_TYPE_WAKEUP) {
198. dbg(DBG_USR2, "HeedM: wakeup\n");
199. // Wake up from sleep state
200. if (sleeping) {
        a. initialize();
201. call Timer.start(TIMER_REPEAT, timer_rate);
        a. sleeping = FALSE;
202. }
203. } else if (pCmdMsg->type == HEED_TYPE_FOCUS) {
204. dbg(DBG_USR2, "HeedM: focus %d\n", pCmdMsg->args.focusaddr);
205. // Cause just one node to chirp and increase its sample rate;
206. // all other nodes stop sending samples (for demo)
207. if (pCmdMsg->args.focusaddr == TOS_LOCAL_ADDRESS) {
208. // OK, we're focusing on me
209. focused = TRUE;
210. call Sounder.init();
```

```
211.  call Timer.stop();
212.  call Timer.start(TIMER_REPEAT, FOCUS_TIMER_RATE);
213.  } else {
214.  // Focusing on someone else
215.  call Timer.stop();
216.  call Timer.start(TIMER_REPEAT, FOCUS_NOTME_TIMER_RATE); }
217.  } else if (pCmdMsg->type == HEED_TYPE_UNFOCUS) {
218.  // Return to normal after focus command
219.  dbg(DBG_USR2, "HeedM: unfocus\n");
220.  focused = FALSE;
221.  call Sounder.stop();
222.  call Timer.stop();
223.  call Timer.start(TIMER_REPEAT, timer_rate);
224.  }
225.  return pMsg;
226.  }
227.  event result_t InterceptHeedMsg.intercept(TOS_MsgPtr msg, void* payload, uint16_t
     payloadLen) {
228.  HeedMsg *sMsg = (HeedMsg *)payload;
229.  #ifdef CLUSTERING_ON
230.  if (call RouteControl.isClusterHead()) {
231.  if (sMsg->type == HEED_TYPE_SENSORREADING) {
232.  numCollectedPoints += sMsg->nPoints;
233.  motesAlive |= sMsg->motesAlive;
234.  } }
235.  #else
236.  if (sMsg->type == HEED_TYPE_SENSORREADING && TOS_LOCAL_ADDRESS == 0)
     {
237.  numCollectedPoints++;
238.  }
239.  #endif
240.  call Leds.yellowToggle();
241.  return SUCCESS;
242.  }
243.  }
```