Хмельницький національний університет
Факультет інформаційних технологій
Кафедра комп'ютерної інженерії та інформаційних систем

КВАЛІФІКАЦІЙНА РОБОТА

Програмно-апаратний комплекс моніторингу віддалених телекомунікаційних вузлів з використанням SNMP.
Назва теми

Рівень вищої освіти *перший (бакалаврський)*

Галузь знань *12 «Інформаційні технології»*
Шифр, назва

Спеціальність *123 «Комп'ютерна інженерія»*
Шифр, назва

Освітня програма *«Комп'ютерна інженерія та програмування»*
Назва

Шифр КвРКІ. 022070.22.04.27

Виконав здобувач IV курсу, група КІ2-22-4 — Підпис — Ілля ПЕЧАЙКО (Ініціали, прізвище)

Керівник — канд.техн.наук, доцент (Науковий ступінь, учене звання) — Підпис — Олексій ІВАНОВ (Ініціали, прізвище)

Нормоконтролер — канд.фіз.-мат.наук, доц. (Науковий ступінь, учене звання) — Підпис — Тетяна КИСІЛЬ (Ініціали, прізвище)

До захисту допускаю:
завідувач кафедри КІПС — Підпис — Ольга ПАВЛОВА (Ініціали, прізвище)
«01» червня 2026 р.
дата

Хмельницький 2026

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА ІНФОРМАЦІЙНИХ СИСТЕМ

Рівень вищої освіти ПЕРШИЙ (БАКАЛАВРСЬКИЙ)

Галузь знань 12 ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

Спеціальність 123 КОМП'ЮТЕРНА ІНЖЕНЕРІЯ

Освітня програма «КОМП'ЮТЕРНА ІНЖЕНЕРІЯ ТА ПРОГРАМУВАННЯ»

ЗАТВЕРДЖУЮ
Завідувачка кафедри КІІС
Ольга ПАВЛОВА

" 10 " 01 2026 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

Печайко Ілля Вікторович
Прізвище, ім'я, по батькові студента

1. Тема проекту (роботи) Програмно-апаратний комплекс моніторингу віддалених телекомунікаційних вузлів з використанням SNMP.

Керівник проекту (роботи) Іванов Олексій Валентинович, к.т.н., доцент.
Прізвище, ім'я, по батькові, науковий ступінь, вчене звання

Затверджена наказом ректора університету від 20.01.2026 р. № 7

2. Термін подання здобувачем роботи на кафедру 01.06.2026 р.

3. Вихідні дані до роботи Завдання на кваліфікаційну роботу

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

Аналіз сучасних технологій, методів та програмних засобів моніторингу стану віддалених телекомунікаційних вузлів у комп'ютерних мережах

Проєктування структури та алгоритму роботи програмно-апаратного комплексу моніторингу віддалених телекомунікаційних вузлів з використанням SNMP

Алгоритмічне та програмне забезпечення розподіленої системи моніторингу

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслень)

Структурна схема комплексу моніторингу

Архітектура оброблення SNMP-даних

Алгоритм SNMP-опитування вузла

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийн

7. Дата видачі завдання « 10 » 01 2026 р.

КАЛЕНДАРНИЙ ПЛАН

№з/п	Назва етапів (розділів) дипломного проекту (роботи)	Термін виконання етапів проекту (роботи)	Примітк
1	Вибір напряму дослідження та узгодження тематики кваліфікаційної роботи з керівником	10.01.2026	виконан
2	Ознайомлення з предметною областю; формулювання мети та задач дослідження; визначення об'єкта та предмета дослідження	01.02.2026	виконан
3	Робота над розділом 1: аналіз технологій моніторингу віддалених телекомунікаційних вузлів та постановка задачі	01.03.2026	виконан
4	Робота над розділом 2: проєктування структури та алгоритму роботи програмно-апаратного комплексу моніторингу з використанням SNMP	01.04.2026	виконан
5	Робота над розділом 3: розроблення алгоритмічного та програмного забезпечення демонстраційного прототипу системи SNMP-моніторингу	29.04.2026	виконано
6	Оформлення пояснювальної записки згідно вимог	25.05.2026	виконано
7	Попередній захист ВКР	26.05.2026	виконано
8	Захист ВКР на засіданні ЕК	Червень 2026 року	

Здобувач _____ Ілля ПЕЧАЙКО
Підпис Ім'я, ПРІЗВИЩЕ

Керівник кваліфікаційної роботи _____ Олексій ІВАНОВ
Підпис Ім'я, ПРІЗВИЩЕ

№ рядка	формат	Позначення	Найменування	Кіл. листів	№ екз	Примітка
			Текстові документи			
1		КвРКІ.022070.22.04.27	Пояснювальна записка	89		
			Графічні матеріали			
2		КвРКІ.022070.22.04.27Е8	Структурна схема	1		
			комплексу моніторингу			
3		КвРКІ.022070.22.04.27Е8	Архітектура оброблення	1		
			SNMP-даних			
4		КвРКІ.022070.22.04.27Е8	Алгоритм	1		
			SNMP-опитування вузла			

Зм	Арк	№ докум	Підпис	Дата
Розробив		Печайко		01.06
Перевір.		Іванов		01.06
Н. контр.		Кисіль		01.06
Затв.		Павлова		01.06

КвРКІ.022070.22.04.27

Відомість проекту

Літера	Аркуш	Аркушів
У	1	1

ХНУ, КІ2-22-4

АНОТАЦІЯ

Тема кваліфікаційної роботи: «Програмно-апаратний комплекс моніторингу віддалених телекомунікаційних вузлів з використанням SNMP».

Автор роботи: Ілля ПЕЧАЙКО.

Керівник роботи: Олексій ІВАНОВ.

Пояснювальна записка: 89 с., 33 рис., 2 табл., 3 дод., 50 джерела.

Графічна частина: 3 креслення.

АРХІТЕКТУРА, БАЗА ДАНИХ, ВЕБІНТЕРФЕЙС, МОНІТОРИНГ, SNMP, ТЕЛЕКОМУНІКАЦІЙНИЙ ВУЗОЛ.

Кваліфікаційна робота бакалавра присвячена проєктуванню програмно-апаратного комплексу моніторингу віддалених телекомунікаційних вузлів з використанням протоколу SNMP. Актуальність теми зумовлена необхідністю контролю стану мережевого обладнання, серверів, джерел безперебійного живлення та інших елементів телекомунікаційної інфраструктури, які можуть розміщуватися віддалено від адміністратора. Доступність вузлів, час їх роботи, стан інтерфейсів, мережевий трафік і події дає змогу швидше виявляти несправності та зменшувати час простою мережевої інфраструктури.

Метою роботи є проєктування програмно-апаратного комплексу для збору, збереження, оброблення та відображення даних моніторингу віддалених телекомунікаційних вузлів. У роботі виконано аналіз технологій моніторингу, розглянуто можливості протоколу SNMP, сформовано вимоги до комплексу, розроблено архітектуру системи, структуру бази даних, алгоритм SNMP-опитування та веб-базований інтерфейс адміністратора.

Підпис здобувача

30.05.2026

Дата

ЗМІСТ

ВСТУП

Актуальність дослідження. Віддалені телекомунікаційні вузли є важливою частиною сучасної мережевої інфраструктури. До їх складу можуть входити маршрутизатори, комутатори, сервери, джерела безперебійного живлення та інше обладнання, яке забезпечує передавання даних, доступ до мережевих сервісів і стабільну роботу інформаційних систем. Такі вузли часто розміщуються поза основним робочим місцем адміністратора, тому їхній стан потрібно контролювати дистанційно. У даному дослідженні ми розглянемо програмно-апаратний комплекс моніторингу віддалених телекомунікаційних вузлів з використанням протоколу SNMP.

Метою дипломної роботи є дослідження особливостей застосування протоколу SNMP та розроблення підходу до побудови програмно-апаратного комплексу моніторингу віддалених телекомунікаційних вузлів. У межах роботи розглядаються механізми збору, оброблення, збереження та відображення інформації про стан мережевого обладнання, а також принципи подання отриманих даних через веб-базований інтерфейс адміністратора. Такий комплекс призначений для контролю доступності обладнання, перегляду базових SNMP-метрик, фіксації подій, визначення стану Online/Offline та забезпечення своєчасного виявлення несправностей у роботі мережевої інфраструктури.

Об’єктом дослідження є процес моніторингу стану віддалених телекомунікаційних вузлів у комп’ютерній мережі.

Предметом дослідження є методи та засоби побудови програмно-апаратного комплексу моніторингу віддалених телекомунікаційних вузлів, що забезпечує збір, оброблення, збереження та відображення даних про стан обладнання з використанням протоколу SNMP і веб-базованого інтерфейсу адміністратора.

1 АНАЛІЗ СУЧАСНИХ ТЕХНОЛОГІЙ, МЕТОДІВ ТА ПРОГРАМНИХ ЗАСОБІВ МОНІТОРИНГУ СТАНУ ВІДДАЛЕНИХ ТЕЛЕКОМУНІКАЦІЙНИХ ВУЗЛІВ У КОМП'ЮТЕРНИХ МЕРЕЖАХ

1.1 Поняття, структура, призначення та особливості функціонування віддалених телекомунікаційних вузлів як об'єктів мережевого моніторингу

Віддалений телекомунікаційний вузол у межах цієї роботи розглядається як сукупність мережевого, серверного та допоміжного обладнання, що знаходиться поза основним робочим місцем адміністратора. До такого вузла можуть входити маршрутизатор, комутатор, сервер, точка доступу, джерело безперебійного живлення або інше обладнання, яке підтримує роботу мережі.

Головна особливість таких вузлів полягає в тому, що адміністратор не завжди має до них швидкий фізичний доступ. Тому важливо мати систему, яка показує стан обладнання, фіксує помилки та допомагає швидше виявити несправність. У сучасних системах моніторингу для цього використовують хости, метрики, тригери, події, графіки та повідомлення адміністратору [1; 4].

Для телекомунікаційних вузлів потрібно контролювати не лише доступність IP-адреси, а й внутрішній стан обладнання, оскільки відповідь на мережевий запит не завжди означає, що пристрій працює стабільно. Вузол може бути доступним у мережі, але при цьому мати перевантажені інтерфейси, помилки передавання даних, нестабільне живлення або інші службові проблеми. До важливих параметрів моніторингу належать час роботи пристрою, стан мережевих інтерфейсів, обсяг вхідного та вихідного трафіку, кількість помилок, температура, стан живлення та наявність аварійних або попереджувальних подій. Підтримка SNMP у RouterOS, Cisco IOS XE, Junos OS, FortiOS, Aruba AOS-CX, Dell OS10 та ExtremeCloud IQ Controller показує, що цей протокол широко застосовується у мережевому обладнанні різних виробників. Завдяки цьому SNMP можна використовувати як універсальну основу для збору службових даних у системах віддаленого моніторингу[27-37].

Рисунок 1.1 – Основні параметри контролю телекомунікаційного вузла

До основних параметрів, які доцільно контролювати під час моніторингу віддаленого телекомунікаційного вузла, належать:

1) доступність вузла в мережі;
2) час безперервної роботи пристрою;
3) назва та ідентифікаційні дані вузла;
4) стан мережевих інтерфейсів;
5) обсяг вхідного та вихідного трафіку;
6) кількість помилок передавання даних;
7) стан резервного живлення;
8) наявність аварійних або попереджувальних подій;
9) час останнього успішного опитування;
10) поточний стан вузла у системі моніторингу.

Аналіз параметрів моніторингу показує, що для оцінювання стану віддаленого телекомунікаційного вузла недостатньо перевіряти лише його доступність. Система моніторингу повинна враховувати як мережеві, так і

службові характеристики пристрою. До таких характеристик належать доступність вузла, час роботи, стан інтерфейсів, інтенсивність мережевого трафіку, кількість помилок передавання даних, стан живлення та події, що виникають у процесі експлуатації обладнання.

1.2 Аналіз сучасних методів моніторингу мережевого обладнання, контролю доступності та оцінювання технічного стану віддалених телекомунікаційних вузлів

Моніторинг мережевого обладнання може виконуватися кількома способами, залежно від типу пристроїв, вимог до точності контролю, доступних протоколів і складності мережевої інфраструктури. Найпростішим способом є ICMP-перевірка доступності, тобто перевірка відповіді пристрою на ping-запит. Такий метод дозволяє швидко визначити, чи доступний вузол у мережі, однак він має обмежені можливості. ICMP показує лише факт наявності або відсутності відповіді від пристрою, але не надає інформації про внутрішній стан обладнання, завантаження інтерфейсів, час безперервної роботи, кількість помилок або інші службові параметри.

Більш інформативним способом є SNMP-опитування. У цьому випадку система моніторингу звертається до SNMP-агента, який працює на мережевому пристрої, і отримує потрібні параметри за визначеними OID. За допомогою SNMP можна отримувати назву пристрою, час його роботи, стан мережевих інтерфейсів, обсяг вхідного та вихідного трафіку, кількість помилок передавання даних та інші показники. SNMP підтримується великою кількістю маршрутизаторів, комутаторів, серверів, джерел безперебійного живлення та іншого обладнання, тому він підходить для моніторингу різнорідної інфраструктури [2; 14; 22].

Окремо використовуються SNMP traps, syslog-повідомлення та агентний моніторинг. SNMP traps відрізняються від звичайного опитування тим, що повідомлення надсилається самим пристроєм у разі виникнення певної події. Наприклад, мережеве обладнання може повідомити про падіння інтерфейсу, зміну стану живлення, перезавантаження або іншу аварійну ситуацію. Такий підхід дозволяє швидше реагувати на події, але потребує налаштування приймача trap-повідомлень у системі моніторингу.

Syslog застосовується для передавання журналів подій від мережевого обладнання або серверів до централізованого сховища. Цей метод зручний для аналізу причин несправностей, оскільки дозволяє переглядати текстові повідомлення про події, помилки, попередження та зміну стану пристроїв. Водночас syslog не завжди підходить для регулярного збору числових метрик, тому його доцільно використовувати як додатковий механізм разом із SNMP.

Агентний моніторинг передбачає встановлення спеціального програмного агента на сервер або робочу станцію. Такий агент може передавати системі моніторингу детальну інформацію про операційну систему, процеси, використання процесора, оперативної пам’яті, дискового простору та мережевих інтерфейсів. Цей підхід зручний для серверів, але не завжди може бути використаний для маршрутизаторів, комутаторів або інших мережевих пристроїв, на які неможливо встановити додаткове програмне забезпечення [3; 11; 12; 16].

Таким чином, кожен метод моніторингу має власне призначення. ICMP доцільно використовувати для базової перевірки доступності, SNMP для отримання службових параметрів обладнання, SNMP traps для оперативного отримання аварійних повідомлень, syslog для аналізу журналів подій, а агентний моніторинг для детального контролю серверів. Для цієї роботи найбільш доцільним є використання SNMP, оскільки він дозволяє отримувати основні параметри стану віддалених телекомунікаційних вузлів і підтримується більшістю мережевого обладнання.

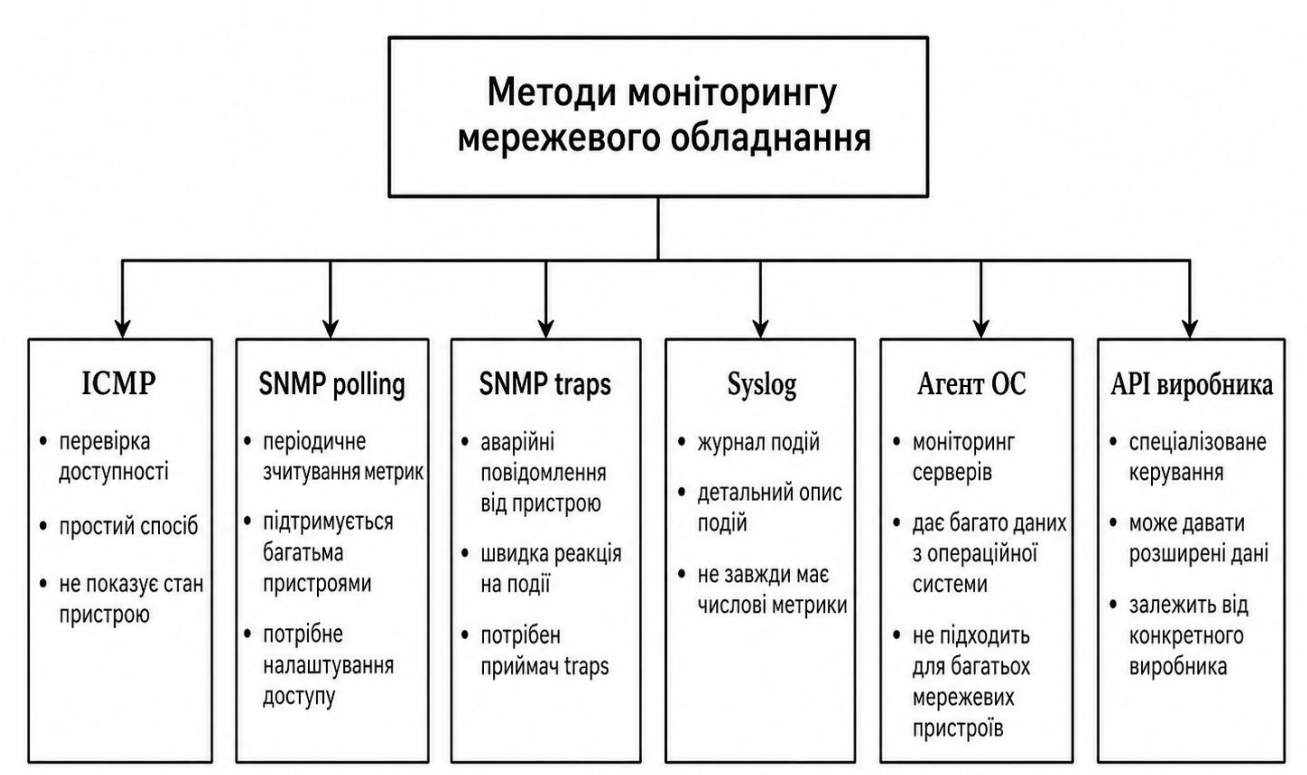


Рисунок 1.2 – Порівняння основних методів моніторингу

1.3 Протокол SNMP як основа збору, передавання та оброблення даних про стан віддалених телекомунікаційних вузлів

SNMP використовується для отримання керуючої, службової та діагностичної інформації від мережевих пристроїв. Його основне призначення полягає у тому, щоб система моніторингу могла віддалено звертатися до обладнання та отримувати від нього дані про поточний стан. До таких даних можуть належати назва пристрою, час його безперервної роботи, стан мережевих інтерфейсів, кількість переданих і прийнятих пакетів, обсяг трафіку, кількість помилок, а також інші параметри, які підтримуються конкретним пристроєм.

У типовій схемі роботи SNMP використовується кілька основних елементів: SNMP manager, SNMP agent, MIB та OID. SNMP manager це система або програмний модуль, який виконує опитування мережевих пристроїв. У межах цієї дипломної роботи роль такого manager може виконувати модуль SNMP-опитування програмно-апаратного комплексу. SNMP agent це

програмний компонент, який працює на самому пристрої або сервері та відповідає на запити manager. Саме agent надає доступ до службових параметрів обладнання.

MIB є базою опису об'єктів, які можуть бути доступні для перегляду через SNMP. Вона визначає структуру параметрів, їхні назви та місце у загальному дереві об'єктів. OID, або об'єктний ідентифікатор, використовується для звернення до конкретного параметра. Наприклад, один OID може відповідати за назву пристрою, інший за час його роботи, ще інший за стан певного мережевого інтерфейсу. Таким чином, SNMP manager не просто звертається до пристрою загалом, а запитує конкретне значення за визначеним OID.

Принцип роботи SNMP можна подати як послідовність простих дій. Спочатку система моніторингу формує запит до пристрою, вказуючи IP-адресу, версію SNMP, параметри доступу та потрібний OID. Далі SNMP agent на пристрої перевіряє запит і, якщо доступ дозволений, повертає відповідне значення. Після цього система моніторингу обробляє отриману відповідь, зберігає її в базі даних або відображає у вебінтерфейсі адміністратора. Якщо відповідь не отримано, система може зафіксувати timeout і змінити стан вузла на Offline.

Для роботи із SNMP у практичних системах можуть використовуватися готові програмні інструменти. Одним із найпоширеніших є Net-SNMP, який надає набір утиліт для роботи із SNMPv1, SNMPv2c та SNMPv3 через IPv4 та IPv6. За допомогою таких інструментів можна виконувати тестові запити до обладнання, переглядати значення OID, перевіряти налаштування доступу та діагностувати роботу SNMP-агента. Для операційних систем Linux роль SNMP-агента може виконувати служба snmpd, яка приймає SNMP-запити та повертає інформацію про систему [22; 23; 24].

У програмній реалізації власного комплексу моніторингу можна використовувати спеціалізовані бібліотеки. Для мови Python одним із таких засобів є PySNMP. Вона дозволяє реалізувати SNMP-запити без використання

сторонніх командних утиліт і підтримує SNMPv1, SNMPv2c та SNMPv3. Завдяки цьому PySNMP може бути використана для створення collector-модуля, який звертається до контрольованих вузлів, отримує значення потрібних OID, обробляє помилки та передає результати до бази даних або іншої частини системи [25; 26].

У практичних системах моніторингу зустрічаються три основні версії SNMP: SNMPv1, SNMPv2c та SNMPv3. SNMPv1 є найстарішою версією протоколу, яка має базові можливості для отримання даних, але обмежена з погляду функціональності та безпеки. SNMPv2c є поширенішою версією, оскільки підтримує зручніші механізми отримання даних, зокрема операцію GetBulk, яка дозволяє ефективніше зчитувати великі обсяги інформації. Проте SNMPv1 і SNMPv2c використовують community string, тому їх небажано застосовувати у відкритих або недостатньо захищених мережах.

SNMPv3 є більш безпечним варіантом протоколу, оскільки підтримує користувачів, автентифікацію та шифрування. Завдяки цьому можна обмежити доступ до SNMP-даних, перевіряти справжність користувача та захищати інформацію під час передавання. Саме SNMPv3 доцільно вважати рекомендованим варіантом для реальних віддалених телекомунікаційних вузлів, особливо якщо моніторинг виконується через незахищені або частково контрольовані канали зв’язку [29; 31; 36].

У межах цієї дипломної роботи SNMP розглядається як основа для збору даних про стан віддалених телекомунікаційних вузлів. Для демонстраційного прототипу допускається використання тестових SNMP-даних або спрощеної моделі опитування, однак сама логіка роботи відповідає реальному підходу: система має перелік вузлів, формує запит, отримує або імітує відповідь, визначає стан вузла, зберігає результат і відображає його у веб-базованому інтерфейсі адміністратора.

Таблиця 1.3 – Порівняння версій SNMP

Версія	Особливості	Рівень захисту	Доцільне використання
SNMPv1	базова версія, community string	низький	старе обладнання або навчальні приклади
SNMPv2c	підтримка GetBulk, community string	низький/середній	закрита мережа, лабораторний стенд
SNMPv3	користувачі, автентифікація, шифрування	вищий	реальні віддалені вузли та захищене середовище

1.4 Огляд сучасних програмних засобів моніторингу, збору SNMP-даних та візуалізації стану віддалених телекомунікаційних вузлів

Для моніторингу мережевої інфраструктури існує багато готових програмних систем, які відрізняються функціональністю, складністю налаштування, способом збирання даних і підходом до візуалізації результатів. Такі засоби використовуються для контролю серверів, мережевого обладнання, каналів зв’язку, служб, баз даних, віртуальних машин та інших елементів інформаційної інфраструктури. У межах цієї роботи доцільно розглянути ті системи, які підтримують SNMP або можуть бути використані разом із SNMP-collector-компонентами.

Однією з найпоширеніших систем моніторингу є Zabbix. Вона підтримує SNMP polling, SNMP traps, елементи даних, тригери, події, шаблони, графіки та дашборди. За допомогою Zabbix можна налаштувати регулярне опитування мережевих пристроїв, отримувати значення потрібних OID, створювати умови спрацювання тригерів і формувати повідомлення про аварійні ситуації. Перевагою Zabbix є широкі можливості та підтримка великої кількості сценаріїв

моніторингу. Водночас для невеликого навчального прототипу така система може бути надмірно складною, оскільки значна частина логіки вже реалізована всередині готової платформи, а метою роботи є саме пояснення принципу побудови власного програмно-апаратного комплексу [1-4].

Grafana використовується переважно для візуалізації даних і побудови інформаційних панелей. Вона дозволяє створювати зручні дашборди, графіки, індикатори стану та панелі для перегляду аварійних подій. Grafana часто застосовується не як самостійний засіб збору даних, а як інструмент для відображення інформації, отриманої з інших систем або баз даних. У контексті цієї роботи Grafana можна розглядати як приклад зручного подання результатів моніторингу адміністратору. Саме ідея веб-базованої панелі з картками стану, таблицями вузлів, графіками та журналом подій може бути використана при проєктуванні демонстраційного інтерфейсу [5-7].

Prometheus є системою збору та зберігання метрик, яка часто використовується в сучасних інфраструктурах. Для роботи із SNMP-пристроями застосовується SNMP Exporter. Його призначення полягає у тому, щоб отримувати дані від SNMP-агентів і перетворювати їх у формат метрик, зрозумілий для Prometheus. Такий підхід добре підходить для масштабованих систем, у яких потрібно зберігати велику кількість часових рядів і будувати графіки зміни параметрів у часі. Проте для простої бакалаврської роботи використання Prometheus і SNMP Exporter може ускладнити реалізацію, тому їх доцільно розглядати як приклад готового промислового підходу [8-10].

Telegraf також може застосовуватися для збору даних моніторингу. Він має SNMP Input Plugin і SNMP Trap Input Plugin, що дозволяє збирати SNMP-метрики та приймати trap-повідомлення від мережевого обладнання. Telegraf часто використовується як проміжний collector-компонент, який отримує дані з різних джерел і передає їх до систем зберігання або візуалізації. Для теми цієї дипломної роботи Telegraf є корисним прикладом того, як може бути організований

окремий модуль збору даних, що працює незалежно від вебінтерфейсу та бази даних [11; 12].

LibreNMS, Checkmk, OpenNMS, NetXMS і PRTG також підтримують SNMP-моніторинг, різні типи сенсорів, обробку подій, візуалізацію та автоматичне виявлення пристроїв. LibreNMS і Checkmk орієнтовані на моніторинг мережевої інфраструктури та можуть автоматично визначати багато параметрів обладнання. OpenNMS і NetXMS також надають засоби контролю подій, доступності та продуктивності. PRTG є прикладом комерційної системи, у якій моніторинг організовано через сенсори, зокрема SNMP-сенсори та приймачі trap-повідомлень [13-21].

Аналіз готових програмних засобів показує, що існує багато рішень для повноцінного моніторингу мережевої інфраструктури. Вони можуть виконувати збір метрик, обробку подій, побудову графіків, надсилання сповіщень і формування звітів. Однак більшість таких систем є достатньо складними для розгортання і налаштування, особливо якщо йдеться про невеликий навчальний прототип. Тому в межах цієї кваліфікаційної роботи доцільно не впроваджувати готову платформу повністю, а використати їх як приклад для формування вимог до власного програмно-апаратного комплексу.

На основі огляду можна зробити висновок, що для демонстраційного прототипу достатньо реалізувати спрощений підхід: перелік контрольованих вузлів, імітацію або отримання базових SNMP-метрик, збереження даних, формування журналу подій і відображення результатів у веб-базованому інтерфейсі. Такий варіант не замінює промислові системи моніторингу, але дозволяє показати основну логіку їх роботи та краще пояснити принцип побудови системи SNMP-моніторингу.

Контрольованих вузлів, базові параметри доступу до них, набір основних метрик, журнал подій і простий вебінтерфейс адміністратора. Такий підхід дозволяє не перевантажувати роботу зайвими функціями, але водночас показати всі ключові етапи роботи системи моніторингу: отримання даних, їх оброблення,

збереження та подання користувачу. Власний демонстраційний прототип у цьому випадку має перевагу перед повним використанням готової системи, оскільки дозволяє самостійно описати логіку роботи окремих компонентів. У готових платформах більшість процесів уже реалізована всередині системи, тому складніше показати, як саме формується запит, як обробляється відповідь, як змінюється стан вузла.

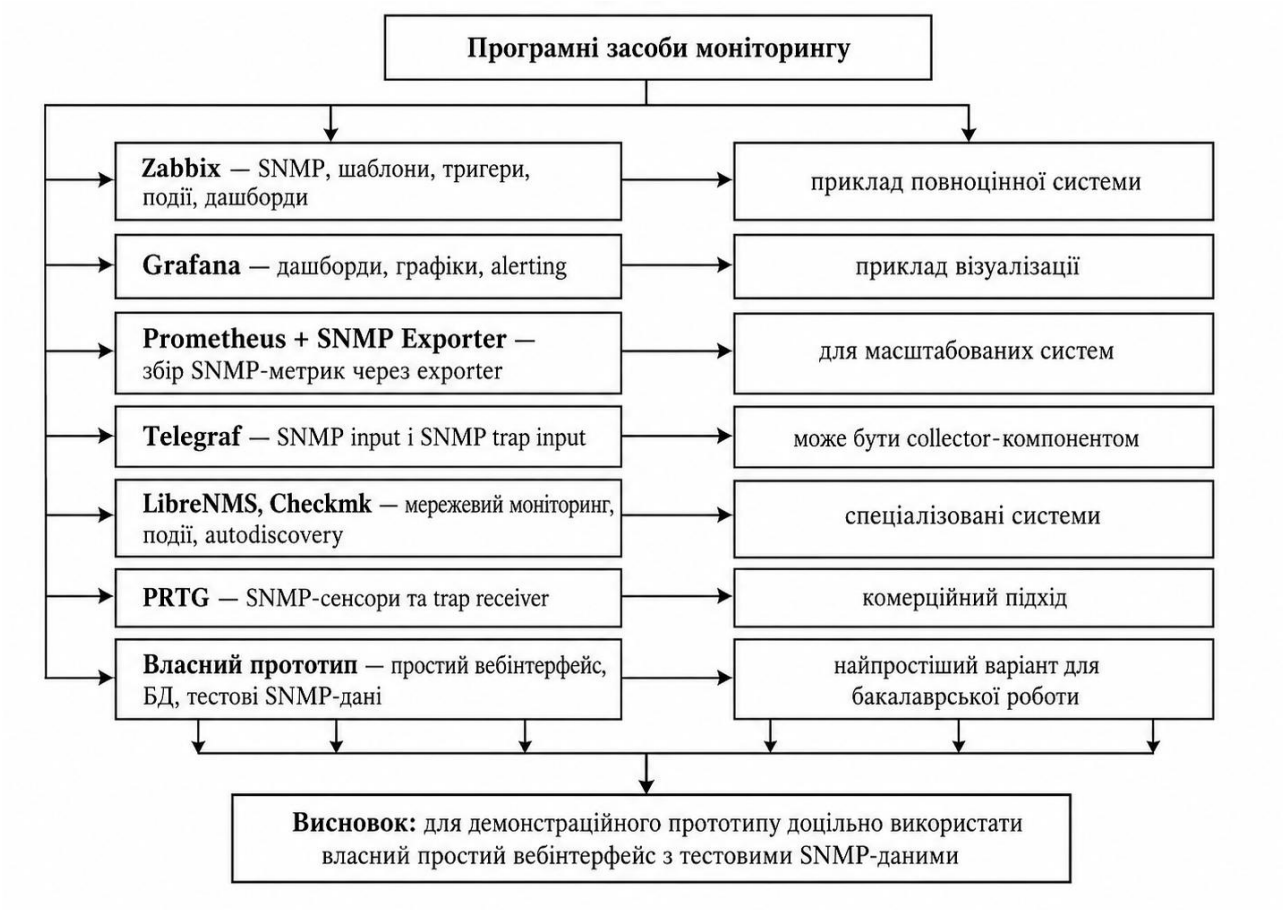


Рисунок 1.4 – Порівняння сучасних засобів моніторингу

1.5 Вимоги інформаційної безпеки до організації SNMP-моніторингу віддалених телекомунікаційних вузлів

Безпека є важливою частиною SNMP-моніторингу, оскільки навіть доступ лише для читання може розкривати службову інформацію про мережеву інфраструктуру. До такої інформації можуть належати назви пристроїв, IP-

адреси, стан інтерфейсів, час роботи обладнання, завантаження каналів, кількість помилок передавання даних та інші параметри. Якщо такі дані потраплять до сторонніх осіб, вони можуть бути використані для аналізу структури мережі, пошуку слабких місць або підготовки несанкціонованого доступу. Тому організація SNMP-моніторингу повинна враховувати не лише зручність отримання даних, а й вимоги інформаційної безпеки.

Одним із головних ризиків є використання стандартних або слабких community string у SNMPv1 та SNMPv2c. Значення на зразок public або private часто використовуються за замовчуванням і можуть бути легко підібрані або виявлені під час сканування мережі. Навіть якщо доступ налаштовано лише у режимі read-only, зловмисник може отримати інформацію про обладнання, інтерфейси, маршрути, навантаження та інші службові параметри. Через це у реальних мережах не рекомендується залишати стандартні community string, а доступ до SNMP потрібно обмежувати лише довіреними адресами.

Для підвищення безпеки SNMP-моніторингу доцільно використовувати фільтрацію за IP-адресами, правила firewall, VPN-доступ або окремий керований сегмент мережі. Сервер моніторингу має бути єдиним вузлом, якому дозволено виконувати SNMP-запити до обладнання. Такий підхід зменшує ризик небажаного опитування пристроїв сторонніми системами. Також бажано не відкривати UDP-порт 161 у публічний інтернет, оскільки це може зробити пристрій доступним для зовнішнього сканування.

Для реального використання більш доцільним є застосування SNMPv3. На відміну від SNMPv1 і SNMPv2c, ця версія підтримує користувачів, автентифікацію та шифрування. Завдяки цьому можна не лише обмежити доступ до пристроїв, а й захистити передавання службових даних. SNMPv3 особливо важливий у випадках, коли моніторинг виконується через віддалені канали зв’язку або мережі, які не повністю контролюються адміністратором.

NIST Cybersecurity Framework 2.0 розглядає моніторинг, контроль доступу та керування ризиками як важливі елементи кібербезпеки. CISA також

наголошує, що інтерфейси керування мережею не повинні бути відкритими у публічний інтернет, оскільки це підвищує ризик несанкціонованого доступу. ENISA у звіті про ландшафт загроз підкреслює зростання складності кіберзагроз, тому системи моніторингу мають враховувати ризики витоку даних, неправильної конфігурації та небажаного доступу до службових інтерфейсів [38-40].

Для проєктованого програмно-апаратного комплексу доцільно передбачити роботу лише у режимі читання. Це означає, що система повинна виконувати отримання параметрів, але не повинна змінювати конфігурацію обладнання через SNMP SET. Відмова від SNMP SET зменшує ризик випадкової або несанкціонованої зміни налаштувань мережевого обладнання. Такий підхід є особливо важливим для навчального або демонстраційного прототипу, де головним завданням є показ логіки моніторингу, а не керування пристроями.

У навчальному середовищі допускається використання SNMPv2c з нестандартним community string, якщо мережа є закритою та не має доступу ззовні. Проте навіть у такому випадку варто дотримуватися базових правил безпеки: не використовувати стандартні значення доступу, не зберігати секрети у відкритому вигляді в коді, обмежувати доступ до SNMP лише з боку сервера моніторингу та вести журнал подій. У реальній системі ці вимоги мають бути посилені шляхом використання SNMPv3, шифрування та контролю доступу.

У демонстраційному прототипі робота виконується з тестовими даними, тому реальні параметри доступу до обладнання не використовуються і не зберігаються. Це дозволяє безпечно показати логіку роботи системи: відображення вузлів, отримання або імітацію SNMP-метрик, визначення стану Online/Offline і формування журналу подій. Водночас архітектура комплексу повинна бути спроєктована так, щоб у майбутньому її можна було адаптувати до роботи з реальними SNMP-пристроями з урахуванням вимог безпеки, обмеження доступу та правильного зберігання параметрів підключення перейти від демонстраційного режиму до практичного використання.

Таблиця 1.5 – Ризики SNMP-моніторингу та способи їх зменшення

Ризик	Можливий наслідок	Спосіб зменшення
SNMPv2c з public/private	несанкціоноване отримання даних	унікальний community string і read-only режим
Відкритий UDP 161	сканування та небажане опитування	firewall і дозвіл лише IP сервера
SNMP через інтернет без захисту	перехоплення або атаки	VPN або SNMPv3
Зберігання секретів у коді	витік параметрів доступу	змінні середовища або захищений файл
Один timeout як аварія	хибне спрацювання	повторні перевірки та журнал подій

1.6 Формування вимог до програмно-апаратного комплексу моніторингу

На основі проведеного аналізу можна сформувати основні вимоги до програмно-апаратного комплексу моніторингу віддалених телекомунікаційних вузлів. Такий комплекс повинен забезпечувати ведення переліку контрольованих пристроїв, відображення їхнього поточного стану, збереження базових метрик, формування журналу подій і надання адміністратору зручного вебінтерфейсу для перегляду результатів моніторингу. Основна увага має приділятися не складному керуванню обладнанням, а своєчасному отриманню інформації про доступність вузлів, зміну їхнього стану та появу можливих несправностей.

До функціональних вимог комплексу доцільно віднести можливість додавання та перегляду телекомунікаційних вузлів, збереження їхніх IP-адрес, назв, версій SNMP і поточного статусу. Також система повинна підтримувати отримання або імітацію базових SNMP-параметрів, зокрема sysName, sysUpTime, стану інтерфейсів і окремих службових значень. Отримані дані

мають зберігатися у базі даних, щоб адміністратор міг переглядати не лише поточний стан, а й історію подій.

Окремою вимогою є формування журналу подій. У ньому повинні фіксуватися успішні опитування вузлів, відсутність відповіді, timeout, перехід пристрою у стан Offline, повернення у стан Online та інші повідомлення, які можуть бути корисними для подальшого аналізу. Наявність такого журналу дозволяє не тільки побачити поточну проблему, а й простежити, коли саме вона виникла та як змінювався стан вузла.

У межах бакалаврської роботи доцільно розглядати не промислову систему масштабу оператора зв'язку, а простий демонстраційний прототип. Для такого прототипу достатньо використати тестові SNMP-дані, структура яких відповідає реальним результатам опитування мережевого обладнання. Це дає можливість показати логіку роботи системи без складного розгортання реального телекомунікаційного стенда, налаштування великої кількості пристроїв і ризику впливу на робочу мережеву інфраструктуру.

Демонстраційний прототип повинен відображати основну ідею майбутнього комплексу: контрольований вузол має певний стан, система отримує або імітує SNMP-відповідь, обробляє результат, записує дані до бази та показує їх у вебінтерфейсі адміністратора. Якщо відповідь від вузла отримана, пристрій може мати стан Online. Якщо відповідь відсутня або виникає timeout, система повинна відобразити стан Offline і створити відповідний запис у журналі подій.

До нефункціональних вимог комплексу можна віднести простоту реалізації, зрозумілість інтерфейсу, модульність, можливість подальшого розширення та безпечний режим роботи. Система не повинна змінювати конфігурацію обладнання через SNMP SET, оскільки в межах цієї роботи достатньо режиму читання. Такий підхід зменшує ризики та відповідає меті демонстраційного прототипу, який має показати принцип моніторингу, а не керування пристроями.

Подальше проєктування має бути пов’язане з трьома основними частинами: вебінтерфейсом, базою даних і алгоритмом SNMP-опитування. У другому розділі потрібно описати структуру програмно-апаратного комплексу, розподіл функцій між компонентами, логіку автономної роботи, базу даних і принципи обробки подій. У третьому розділі доцільно розглянути алгоритмічне та програмне забезпечення демонстраційного прототипу, описати псевдокод основних процедур, структуру програмного забезпечення, веб-базований інтерфейс і приклади застосування системи для моніторингу віддалених телекомунікаційних вузлів.

Рисунок 1.6 – Функціональні вимоги до комплексу

1.7 Обґрунтування вибору підходу до побудови програмно-апаратного комплексу моніторингу віддалених телекомунікаційних вузлів з використанням SNMP та постановка задачі дослідження

Проведений аналіз показує, що SNMP є доцільною основою для моніторингу віддалених телекомунікаційних вузлів. Його перевага полягає у широкій підтримці мережевим обладнанням, можливості отримання стандартних параметрів через OID та використанні різних версій протоколу залежно від рівня безпеки.

Для цієї кваліфікаційної роботи обрано спрощений підхід: проєктується програмно-апаратний комплекс і створюється демонстраційний прототип вебінтерфейсу з тестовими SNMP-даними. Такий варіант дозволяє показати основні принципи системи: перелік вузлів, стан Online/Offline, базові метрики, журнал подій і структуру зберігання даних.

Для вирішення поставленого завдання необхідно виконати такі етапи:

1) проаналізувати особливості віддалених телекомунікаційних вузлів;

2) розглянути основні методи моніторингу мережевого обладнання;

3) дослідити принципи роботи SNMP та відмінності між версіями протоколу;

4) проаналізувати готові системи моніторингу та їх можливості;

5) визначити вимоги до програмно-апаратного комплексу;

6) спроєктувати структурну схему комплексу моніторингу;

7) розробити структуру бази даних для вузлів, метрик і подій;

8) описати алгоритм SNMP-опитування та обробки подій;

9) створити демонстраційний прототип вебінтерфейсу адміністратора;

10) провести тестування прототипу на основі тестових SNMP-даних.

У першому розділі розглянуто особливості віддалених телекомунікаційних вузлів і визначено, що для їх контролю важливо відстежувати доступність, час роботи, стан інтерфейсів, трафік, помилки та події.

Проаналізовано основні методи моніторингу мережевого обладнання та віддалених телекомунікаційних вузлів, серед яких ICMP, SNMP polling, SNMP traps, syslog, агентний моніторинг і API виробників. У процесі аналізу встановлено, що кожен із цих методів має власне призначення, переваги та

обмеження. Найпростішим способом контролю доступності вузла є ICMP-перевірка, яка дозволяє швидко визначити, чи відповідає пристрій у мережі. Однак такий підхід не дає інформації про внутрішній стан обладнання, стан інтерфейсів, трафік, час безперервної роботи або кількість помилок. Тому ICMP може використовуватися лише як базовий допоміжний засіб перевірки доступності.

Більш інформативним методом є SNMP polling, який передбачає періодичне зчитування параметрів із мережевого обладнання за допомогою SNMP-запитів. Такий підхід дозволяє отримувати службову інформацію про пристрій, зокрема його назву, час роботи, стан інтерфейсів, обсяг переданих і прийнятих даних, а також інші параметри, що зберігаються у відповідних MIB-структурах. Завдяки цьому SNMP polling є одним із найбільш доцільних методів для побудови системи моніторингу віддалених телекомунікаційних вузлів.

Окремо було розглянуто SNMP traps, які відрізняються від звичайного періодичного опитування тим, що ініціатором повідомлення виступає сам пристрій. У разі виникнення певної події, наприклад відмови інтерфейсу, перезавантаження обладнання або проблеми з живленням, SNMP-агент може самостійно надіслати повідомлення системі моніторингу. Такий механізм корисний для швидкого реагування на аварійні ситуації, але потребує налаштування приймача trap-повідомлень і не замінює повністю регулярне опитування вузлів.

Метод syslog також має важливе значення для адміністрування мережевої інфраструктури. Він дозволяє передавати журнали подій від пристроїв до централізованої системи зберігання або аналізу. Проте syslog більше орієнтований на текстові повідомлення про події, ніж на регулярне отримання числових метрик. Через це його доцільно розглядати як додатковий засіб, який може доповнювати SNMP-моніторинг, але не повністю замінювати його.

Агентний моніторинг є зручним для серверів і робочих станцій, оскільки дає змогу отримувати розширену інформацію про операційну систему, процеси,

використання процесора, пам’яті та дискового простору. Водночас такий підхід не завжди підходить для маршрутизаторів, комутаторів, джерел безперебійного живлення та іншого мережевого обладнання, на яке неможливо встановити окремий програмний агент. API виробників може надавати розширені можливості керування та моніторингу, але часто залежить від конкретного обладнання і не завжди є універсальним рішенням.

У результаті порівняння методів моніторингу встановлено, що саме SNMP є найбільш доцільним варіантом для цієї кваліфікаційної роботи. Це пояснюється тим, що SNMP підтримується великою кількістю мережевих пристроїв різних виробників і дозволяє отримувати типові службові параметри стану обладнання. За допомогою SNMP можна контролювати доступність пристрою, його назву, час безперервної роботи, стан мережевих інтерфейсів, обсяг вхідного та вихідного трафіку, кількість помилок передавання даних і деякі параметри живлення. Саме ці характеристики є важливими для контролю віддалених телекомунікаційних вузлів.

Розглянуто основні версії протоколу SNMP: SNMPv1, SNMPv2c та SNMPv3. Було визначено, що SNMPv1 є найстарішою версією протоколу і має обмежені можливості з погляду безпеки та функціональності. SNMPv2c є поширенішою версією, яка підтримує зручніші механізми отримання даних, зокрема операцію GetBulk, однак також використовує community string, що не забезпечує достатнього рівня захисту у відкритих мережах. Тому SNMPv1 і SNMPv2c доцільно застосовувати лише у контрольованому середовищі, наприклад у лабораторній мережі або навчальному стенді.

Для реального використання більш доцільним є застосування SNMPv3, оскільки ця версія підтримує автентифікацію користувачів і шифрування даних. Це особливо важливо для віддалених телекомунікаційних вузлів, доступ до яких може здійснюватися через незахищені або частково контрольовані канали зв’язку. У межах цієї роботи зроблено висновок, що для демонстраційного прототипу можна використовувати тестові SNMP-дані або умовний режим

SNMPv2c read-only, але для реального впровадження системи бажано орієнтуватися саме на SNMPv3.

Також було виконано огляд сучасних програмних засобів моніторингу, серед яких Zabbix, Grafana, Prometheus, Telegraf, LibreNMS, Checkmk, OpenNMS, NetXMS і PRTG. Кожна з цих систем має власну сферу застосування. Zabbix є прикладом повноцінної системи моніторингу з підтримкою SNMP, тригерів, подій, шаблонів і дашбордів. Grafana переважно використовується для візуалізації даних. і побудови графіків. Prometheus у поєднанні зі SNMP Exporter може застосовуватися для збору SNMP-метрик у масштабованих системах. Telegraf може виконувати роль collector-компонента для збору метрик і передавання їх до інших систем.

LibreNMS, Checkmk, OpenNMS і NetXMS є спеціалізованими системами мережевого моніторингу, які підтримують автоматичне виявлення пристроїв, обробку подій і роботу з SNMP. PRTG є прикладом комерційного підходу до моніторингу, де велика частина функціоналу подається через готові сенсори та зручний інтерфейс. Аналіз цих засобів показав, що готові платформи мають широкі можливості, але для бакалаврської роботи вони можуть бути надмірно складними, якщо метою є не впровадження готового продукту, а пояснення принципів побудови власного програмно-апаратного комплексу.

На основі проведеного огляду обґрунтовано доцільність створення спрощеного демонстраційного прототипу. Такий прототип не повинен конкурувати з промисловими системами моніторингу, але має показати основну логіку їх роботи. До цієї логіки належать ведення переліку контрольованих вузлів, отримання або імітація SNMP-метрик, визначення стану Online/Offline, запис подій, збереження даних і відображення результатів у веб-базованому інтерфейсі адміністратора. Саме такий підхід відповідає рівню бакалаврської роботи і дозволяє продемонструвати розуміння предметної області без надмірного ускладнення реалізації.

Сформовано основні вимоги до подальшого проєктування програмно-апаратного комплексу. Система повинна забезпечувати відображення списку вузлів, збереження інформації про пристрої, показ поточного стану, зберігання базових метрик, формування журналу подій і подання результатів у зручному для адміністратора вигляді. Також важливо, щоб комплекс працював у безпечному режимі читання і не виконував зміну конфігурації обладнання через SNMP SET. Такий підхід зменшує ризики для реальної мережевої інфраструктури та робить систему придатною для навчального застосування.

Окремо визначено, що для демонстраційного прототипу допустимо використовувати тестові SNMP-дані, структура яких відповідає реальним результатам опитування обладнання. Це дозволяє перевірити логіку роботи вебінтерфейсу, журналу подій, таблиці вузлів і відображення метрик без підключення до реального телекомунікаційного обладнання. Такий варіант є практичним для умов виконання кваліфікаційної роботи, оскільки не потребує складного лабораторного стенда, але дає можливість показати принцип роботи майбутньої системи.

Проведений аналіз також показав, що важливо не обмежуватися лише поточним станом вузла. Для повноцінного моніторингу потрібно мати можливість зберігати історію подій і результатів перевірок. Навіть у спрощеному прототипі журнал подій має значення, оскільки дозволяє побачити, коли вузол був доступний, коли виник timeout, коли пристрій перейшов у стан Offline і коли доступність була відновлена. Це робить систему більш корисною для адміністратора та наближує її до реальних засобів моніторингу.

Крім фіксації поточних станів, історія подій дозволяє виконувати простий аналіз стабільності роботи вузлів. Наприклад, якщо певний пристрій часто переходить у стан Offline або регулярно має timeout, це може свідчити про проблеми з каналом зв’язку, неправильні налаштування SNMP, перевантаження обладнання або нестабільне живлення. У такому випадку адміністратор може не лише побачити факт несправності, а й оцінити її повторюваність. Збереження

історії перевірок також корисне для порівняння роботи різних типів обладнання. Маршрутизатор, комутатор, сервер або джерело безперебійного живлення можуть мати різні параметри контролю, однак для системи моніторингу важливо зберігати їх у єдиній структурі. Це спрощує подальше відображення даних у вебінтерфейсі, формування графіків, перегляд журналу подій і підготовку коротких звітів про стан мережевої інфраструктури.

1.8 Постановки задачі

На основі проведеного аналізу технологій моніторингу, методів збору даних, можливостей протоколу SNMP та сучасних програмних засобів моніторингу можна сформулювати постановку задачі дослідження. Основна задача кваліфікаційної роботи полягає у проєктуванні програмно-апаратного комплексу моніторингу віддалених телекомунікаційних вузлів з використанням протоколу SNMP та описі демонстраційного прототипу, який показує основну логіку роботи такої системи.

У межах роботи необхідно розглянути віддалений телекомунікаційний вузол як об’єкт моніторингу, визначити його основні складові та параметри, які потрібно контролювати. До таких параметрів належать доступність вузла, час безперервної роботи пристрою, стан мережевих інтерфейсів, обсяг вхідного та вихідного трафіку, кількість помилок, стан живлення та наявність аварійних або попереджувальних подій.

Також необхідно обґрунтувати використання протоколу SNMP як основного засобу збору службових даних з мережевого обладнання. Для цього потрібно врахувати принцип роботи SNMP manager, SNMP agent, MIB та OID, а також особливості використання різних версій протоколу. Окрему увагу слід приділити питанням безпеки, оскільки навіть доступ лише для читання може розкривати службову інформацію про мережеву інфраструктуру.

Поставлена задача передбачає проєктування структури програмно-апаратного комплексу, у якій будуть виділені основні компоненти: контрольовані телекомунікаційні вузли, модуль SNMP-опитування, модуль

оброблення результатів, база даних, журнал подій та веб-базований інтерфейс адміністратора. Такий поділ дозволяє логічно розмежувати функції системи та показати повний шлях проходження даних від вузла до користувача.

У межах демонстраційного прототипу доцільно використовувати тестові SNMP-дані, структура яких відповідає реальним результатам опитування мережевого обладнання. Це дозволяє перевірити логіку роботи системи без розгортання складного лабораторного стенда та без ризику впливу на реальну мережеву інфраструктуру. При цьому архітектура комплексу повинна залишатися придатною для подальшого переходу до роботи з реальними SNMP-пристроями.

Для досягнення поставленої мети необхідно виконати такі завдання:

1) проаналізувати особливості віддалених телекомунікаційних вузлів як об’єктів моніторингу;

2) розглянути основні методи моніторингу мережевого обладнання;

3) дослідити принципи роботи протоколу SNMP та його роль у зборі службових даних;

4) проаналізувати сучасні програмні засоби моніторингу, які підтримують SNMP або можуть використовуватися разом із ним;

5) визначити основні вимоги до програмно-апаратного комплексу моніторингу;

6) спроєктувати структуру комплексу та розподілити функції між його компонентами;

7) описати логіку збереження даних, формування метрик і журналу подій;

8) розробити алгоритмічну логіку SNMP-опитування та оброблення результатів;

9) описати веб-базований інтерфейс адміністратора для перегляду стану вузлів;

10) навести практичні приклади застосування системи для моніторингу маршрутизатора, комутатора, сервера та джерела безперебійного живлення.

Таким чином, постановка задачі дослідження полягає у створенні обґрунтованої структури програмно-апаратного комплексу моніторингу, який забезпечує отримання, оброблення, збереження та відображення даних про стан віддалених телекомунікаційних вузлів. У подальших розділах ці положення використовуються для проєктування архітектури комплексу та опису демонстраційного прототипу системи SNMP-моніторингу.

1.9 Висновки до розділу 1

У першому розділі було розглянуто сучасні технології, методи та програмні засоби моніторингу віддалених телекомунікаційних вузлів. Визначено, що такі вузли можуть містити маршрутизатори, комутатори, сервери, точки доступу, джерела безперебійного живлення та інше обладнання, яке забезпечує стабільну роботу мережевої інфраструктури.

Проаналізовано основні параметри, які доцільно контролювати під час моніторингу телекомунікаційного вузла. До них належать доступність пристрою, час безперервної роботи, стан мережевих інтерфейсів, обсяг вхідного та вихідного трафіку, кількість помилок передавання даних, стан живлення та події, що виникають у процесі експлуатації обладнання.

Розглянуто основні методи моніторингу мережевого обладнання, зокрема ICMP-перевірку доступності, SNMP polling, SNMP traps, syslog-повідомлення, агентний моніторинг та API виробників. Встановлено, що ICMP підходить лише для базової перевірки доступності, тоді як SNMP дозволяє отримувати більш детальну службову інформацію про стан обладнання.

Окремо досліджено протокол SNMP як основу збору даних у системах моніторингу. Визначено роль SNMP manager, SNMP agent, MIB та OID у процесі отримання параметрів від мережевих пристроїв. Також розглянуто особливості SNMPv1, SNMPv2c та SNMPv3. Для реального використання доцільно орієнтуватися на SNMPv3, оскільки ця версія підтримує автентифікацію та шифрування.

Виконано огляд сучасних програмних засобів моніторингу, серед яких Zabbix, Grafana, Prometheus, Telegraf, LibreNMS, Checkmk, OpenNMS, NetXMS і PRTG. Встановлено, що готові системи мають широкі можливості, але для бакалаврської роботи доцільно розглядати спрощений демонстраційний прототип, який показує основну логіку SNMP-моніторингу без надмірного ускладнення реалізації.

Розглянуто вимоги інформаційної безпеки до організації SNMP-моніторингу. Визначено, що система повинна працювати у режимі читання, не використовувати SNMP SET, обмежувати доступ до SNMP лише дозволеними адресами та не зберігати параметри доступу у відкритому вигляді. Для демонстраційного прототипу допустимо використовувати тестові SNMP-дані, що дозволяє безпечно показати логіку роботи комплексу.

На основі проведеного аналізу сформовано вимоги до програмно-апаратного комплексу моніторингу. Система повинна забезпечувати ведення переліку вузлів, отримання або імітацію SNMP-метрик, визначення стану Online/Offline, збереження даних, формування журналу подій і відображення результатів у веб-базованому інтерфейсі адміністратора.

У результаті проведеного аналізу було сформульовано постановку задачі дослідження, яка полягає у проєктуванні програмно-апаратного комплексу моніторингу віддалених телекомунікаційних вузлів з використанням протоколу SNMP. Подальше проєктування має бути спрямоване на розроблення загальної структури комплексу, визначення ролей його основних компонентів, побудову логіки збереження та оброблення даних, опис алгоритму SNMP-опитування, формування журналу подій і створення демонстраційного прототипу системи моніторингу. Такий підхід дозволяє перейти від аналізу існуючих технологій до практичного опису системи, яка може відображати стан вузлів.

2 ПРОЄКТУВАННЯ СТРУКТУРИ ТА АЛГОРИТМУ РОБОТИ ПРОГРАМНО-АПАРАТНОГО КОМПЛЕКСУ МОНІТОРИНГУ ВІДДАЛЕНИХ ТЕЛЕКОМУНІКАЦІЙНИХ ВУЗЛІВ З ВИКОРИСТАННЯМ SNMP

2.1 Структура програмно-апаратного комплексу моніторингу віддалених вузлів

Архітектура програмно-апаратного комплексу моніторингу побудована навколо ідеї розділення джерел даних, логіки збору інформації, засобів зберігання та засобів відображення результатів. Такий підхід дозволяє не змішувати всі функції в одному умовному модулі, а поділити систему на окремі частини, кожна з яких виконує свою роль. У звичайній невеликій мережі адміністратор часто перевіряє обладнання вручну: відкриває вебпанель маршрутизатора, виконує ping-запит, переглядає журнали подій або підключається до окремого сервера. Такий спосіб може бути прийнятним, поки кількість вузлів невелика і всі пристрої знаходяться поруч. Якщо ж обладнання розміщене віддалено, а вузли мають різне призначення, ручна перевірка стає незручною та займає більше часу.

У проєктованій системі джерелами даних є віддалені телекомунікаційні вузли. До них можуть належати маршрутизатори, комутатори, сервери, точки доступу, контролери або джерела безперебійного живлення. Кожен із таких вузлів може мати власні параметри контролю: доступність у мережі, час безперервної роботи, стан інтерфейсів, обсяг трафіку, кількість помилок або стан живлення. Для дипломного прототипу достатньо використати умовний набір вузлів, оскільки головна задача полягає не у створенні промислової системи, а у демонстрації логіки контролю стану обладнання. При цьому обрана структура не суперечить реальному застосуванню, оскільки у справжній мережі такі самі блоки можуть бути підключені до реальних SNMP-пристроїв.

Центральним елементом комплексу є модуль SNMP-опитування. Він виконує роль SNMP manager, тобто формує запити до контрольованих вузлів, звертається до SNMP-агентів на пристроях, очікує відповідь і передає отримані результати на подальшу обробку. У найпростішому варіанті такий модуль може працювати послідовно, перевіряючи вузли один за одним. Однак сама архітектура не обмежує систему лише послідовним способом роботи. За потреби запити до різних вузлів можна виконувати паралельно, оскільки результат опитування одного пристрою майже не залежить від результатів інших пристроїв. Це особливо важливо для віддалених вузлів, де один пристрій може відповідати швидко, а інший із затримкою або не відповідати взагалі.

Окремо в архітектурі доцільно виділити модуль тестових даних. Його наявність важлива саме для демонстраційного прототипу. На практиці не завжди є можливість підключити реальне телекомунікаційне обладнання або дозволити сторонній програмі виконувати запити до робочої мережі. Тестовий режим дозволяє показати роботу вебінтерфейсу, журналу подій, бази даних і логіки формування станів Online/Offline без впливу на реальні пристрої. Такий підхід не замінює повноцінне SNMP-опитування, але дає можливість перевірити структуру майбутньої системи та продемонструвати її принцип роботи.

База даних у системі виконує роль єдиного місця зберігання поточного стану вузлів і короткої історії моніторингу. У ній доцільно зберігати перелік контрольованих пристроїв, IP-адреси, версію SNMP, час останньої перевірки, поточний статус, значення основних метрик і записи журналу подій. Для навчального прототипу достатньо використати SQLite, оскільки така база не потребує встановлення окремого сервера і може зберігатися в одному файлі. У разі подальшого розвитку комплексу SQLite можна замінити на PostgreSQL або іншу серверну систему керування базами даних без принципової зміни загальної логіки роботи.

Журнал подій не варто розглядати як другорядний елемент системи. Для адміністратора сам факт отримання окремої метрики часто менш важливий, ніж

розуміння того, коли саме вузол став недоступним, коли відновився зв’язок, скільки разів виникав timeout і які події відбувалися під час роботи системи. Тому журнал подій є окремим компонентом логічної архітектури. Він може зберігатися в тій самій базі даних, що й метрики, але його призначення відрізняється: журнал відповідає не за числові значення, а за фіксацію подій, зміну станів і повідомлення, які мають значення для адміністратора.

Вебінтерфейс адміністратора є верхнім рівнем системи. Він не повинен безпосередньо опитувати мережеві пристрої, оскільки це зробило б його залежним від затримок, помилок і особливостей SNMP-запитів. Його основне завдання полягає у відображенні вже підготовлених даних: списку вузлів, поточних статусів, останніх метрик, графіків і повідомлень журналу подій. Завдяки цьому браузер користувача не залежить від особливостей роботи SNMP, а адміністратор отримує зрозумілу панель моніторингу замість набору команд, конфігураційних файлів або окремих сторінок різних пристроїв.

Загальна архітектура комплексу наведена на рисунку 2.1. Вона показує, що контрольовані вузли не взаємодіють безпосередньо з адміністратором. Усі дані проходять через модуль опитування, після чого обробляються, зберігаються в базі даних і відображаються у вебінтерфейсі. Така побудова зменшує кількість ручних дій, робить систему більш керованою та дозволяє надалі розширювати її без повної перебудови архітектури. Наприклад, у майбутньому можна додати нові типи вузлів, додаткові SNMP-метрики, сповіщення або механізм авторизації користувачів, не змінюючи основний принцип роботи комплексу.

У запропонованій схемі адміністратор працює не з кожним пристроєм окремо, а з єдиною системою моніторингу. Це зменшує кількість ручних дій, оскільки не потрібно окремо відкривати вебпанель маршрутизатора, виконувати ping-запити, перевіряти журнали або вручну переглядати стан кожного вузла. Система збирає основну інформацію централізовано і подає її в однаковому форматі, незалежно від типу контрольованого обладнання.

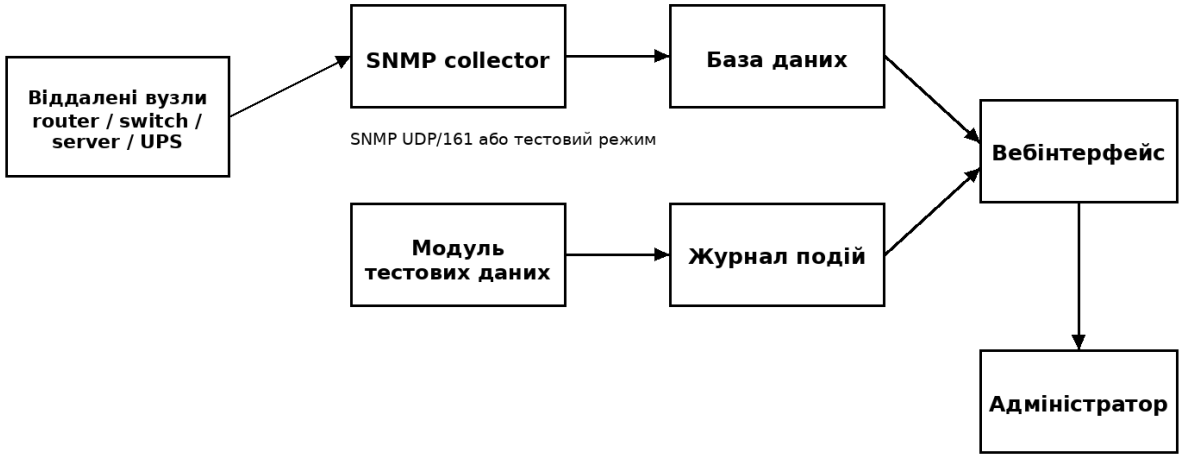


Рисунок 2.1 – Загальна архітектура розподіленої системи моніторингу

Після розгляду загальної схеми потрібно уточнити роль кожного компонента програмно-апаратного комплексу. Це необхідно для того, щоб архітектура системи не виглядала як набір окремих блоків без пояснення їхнього призначення. Кожен компонент у такій системі виконує власну функцію: одні елементи є джерелами даних, інші відповідають за отримання інформації, обробку результатів, збереження даних або їх відображення користувачу. Саме розподіл ролей дозволяє зробити систему більш зрозумілою та придатною для подальшого розширення.

Віддалений телекомунікаційний вузол у цій роботі не є активним учасником керування системою. Він не приймає рішень щодо зміни власного стану і не запускає процес моніторингу самостійно. Його роль полягає у наданні службових даних у відповідь на коректний SNMP-запит. Якщо пристрій має налаштований SNMP-agent і дозволяє доступ із боку системи моніторингу, він може повернути значення потрібних параметрів. До таких параметрів можуть належати sysName, sysUpTime, стан інтерфейсів, лічильники трафіку або інші показники, які підтримуються конкретним обладнанням.

Важливим моментом є те, що проєктована система не змінює конфігурацію віддаленого вузла. Вона не виконує SNMP SET, не змінює параметри

інтерфейсів, не перезапускає служби і не втручається в роботу обладнання. Система працює у режимі читання, що є більш безпечним для навчального, тестового та демонстраційного використання. Такий підхід дозволяє отримати інформацію про стан вузла, але не створює ризику випадкової зміни налаштувань мережевого обладнання.

SNMP collector можна умовно поділити на кілька внутрішніх частин: формування запиту, надсилання запиту до вузла, очікування відповіді, перевірка помилок, обробка отриманого значення та передавання результату іншим компонентам системи. На першому етапі collector визначає, який вузол потрібно опитати, яку версію SNMP використати та який OID потрібно зчитати. Далі формується запит до SNMP-agent пристрою. Якщо відповідь отримано, collector передає значення до модуля обробки або одразу до бази даних. Якщо відповідь не отримано, система повинна зафіксувати помилку.

Якщо під час опитування виникає timeout, collector не повинен просто завершувати роботу з помилкою. У реальній системі моніторингу така ситуація є важливою подією, яку потрібно передати далі. Наприклад, інформація про timeout може бути передана до модуля аналізу або безпосередньо до журналу подій. Після цього система може відобразити вузол як недоступний або перевести його у стан Offline. Така поведінка є більш правильною, оскільки адміністратор бачить не лише відсутність даних, а конкретну причину зміни стану вузла.

Модуль аналізу в простому демонстраційному прототипі може бути частиною SNMP collector. Однак на рівні архітектури його доцільно виділяти окремо, оскільки він виконує іншу логічну роль. Collector відповідає за отримання даних, а модуль аналізу за їх інтерпретацію. Саме цей модуль визначає, який статус потрібно присвоїти вузлу, чи змінився його стан порівняно з попереднім опитуванням, чи потрібно створити подію в журналі та чи потрібно показати попередження у вебінтерфейсі.

Наприклад, якщо вузол раніше мав стан Online, але під час наступного опитування не відповів, система не повинна просто залишити порожнє поле в таблиці. Вона має сформувати зрозумілий результат: зафіксувати timeout, оновити час перевірки, змінити статус вузла на Offline або Warning та додати запис у журнал подій. Якщо під час наступного циклу опитування вузол знову відповість, система повинна відобразити відновлення доступності. Такий підхід робить моніторинг більш корисним для адміністратора.

Універсальність архітектури забезпечується не великою кількістю функцій, а правильним розділенням ролей між компонентами. Якщо у майбутньому потрібно додати інший тип вузла, наприклад мережеве сховище, точку доступу або UPS, достатньо описати його параметри та додати потрібні OID до переліку контрольованих метрик. При цьому не потрібно змінювати всю систему повністю. Collector і надалі виконуватиме опитування, база даних зберігатиме результати, а вебінтерфейс відображатиме підготовлену інформацію.

Так само, якщо потрібно змінити спосіб подання результатів, можна переробити вебінтерфейс без зміни логіки SNMP-опитування. Наприклад, замість простої таблиці можна додати графіки, картки стану або сторінку конкретного вузла. При цьому структура збору даних і журнал подій можуть залишатися без змін. Саме таке розділення робить систему зручною для подальшого розвитку та дозволяє поступово розширювати функціональність без повної перебудови архітектури.

Рисунок 2.2 деталізує взаємодію компонентів комплексу. На ньому показано, що між плануванням опитування, формуванням SNMP-запиту, обробкою відповіді, контролем timeout і записом до бази даних існує послідовний логічний зв’язок. У реальній програмі ці частини можуть бути реалізовані як окремі функції, класи або модулі, однак у пояснювальній записці важливо показати саме логіку взаємодії між ними. Завдяки цьому стає зрозуміло, як дані проходять шлях від віддаленого вузла до вебінтерфейсу адміністратора.

Таким чином, уточнення ролей компонентів дозволяє краще пояснити принцип роботи всієї системи. Віддалені вузли надають дані, SNMP collector виконує опитування, модуль аналізу визначає стан обладнання, база даних зберігає результати, журнал подій фіксує важливі зміни, а вебінтерфейс показує адміністратору підготовлену інформацію. Така структура є достатньо простою для демонстраційного прототипу, але водночас логічною для подальшого розвитку системи моніторингу.

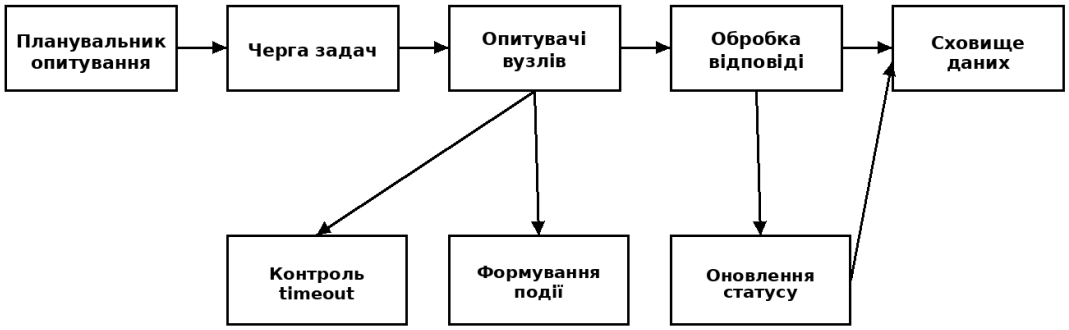


Рисунок 2.2 – Схема взаємодії основних компонентів комплексу

У межах дипломної роботи архітектура не повинна бути надмірно складною. Інакше основна ідея SNMP-моніторингу губиться за великою кількістю допоміжних модулів. Тому для другого розділу достатньо пояснити ті блоки, які реально потрібні для демонстраційного прототипу: вузли, collector, база, журнал, вебінтерфейс і тестовий режим. Цього набору досить, щоб показати повний шлях даних від пристрою до адміністратора.

Важливо також врахувати, що віддалені вузли можуть мати різну якість каналу зв’язку. Один пристрій може відповідати швидко, інший - із затримкою, а третій час від часу бути недоступним. Через це система повинна працювати з timeout, повторними перевірками та журналом помилок. Якщо цього не передбачити на етапі архітектури, то навіть простий прототип буде виглядати непереконливо.

На рівні даних кожен вузол повинен мати мінімальний опис: назву, IP-адресу, версію SNMP, поточний статус і час останнього опитування. Для більш

повної системи до цього додаються місце розташування, тип обладнання, профіль OID і відповідальна особа. У бакалаврській роботі можна обмежитися базовими полями, але потрібно залишити можливість розширення структури.

Побудована архітектура відповідає задачі створення не промислового продукту, а зрозумілого прототипу. Її перевага полягає в тому, що вона пояснює реальний принцип моніторингу без необхідності піднімати велику платформу на зразок Zabbix або Prometheus. Для навчальної роботи це прийнятний компроміс між простотою і технічною логікою.

Окремої уваги потребує зв’язок між базою даних і вебінтерфейсом. Інтерфейс не має напряму звертатися до вузлів, оскільки тоді він стає залежним від мережевих затримок і помилок SNMP. Правильніше, щоб вебсторінка читала вже оброблені дані. Це дозволяє відкрити панель навіть тоді, коли частина вузлів тимчасово не відповідає. У такому разі користувач бачить останній відомий стан і подію про помилку.

Загалом запропонована архітектура має три рівні. Перший рівень - джерела даних, тобто телекомунікаційні вузли. Другий рівень - внутрішня логіка комплексу, до якої входять опитування, аналіз, база даних і журнал подій. Третій рівень - взаємодія з адміністратором через вебінтерфейс. Такий поділ добре підходить для подальшого опису функцій системи у підрозділі 2.2.

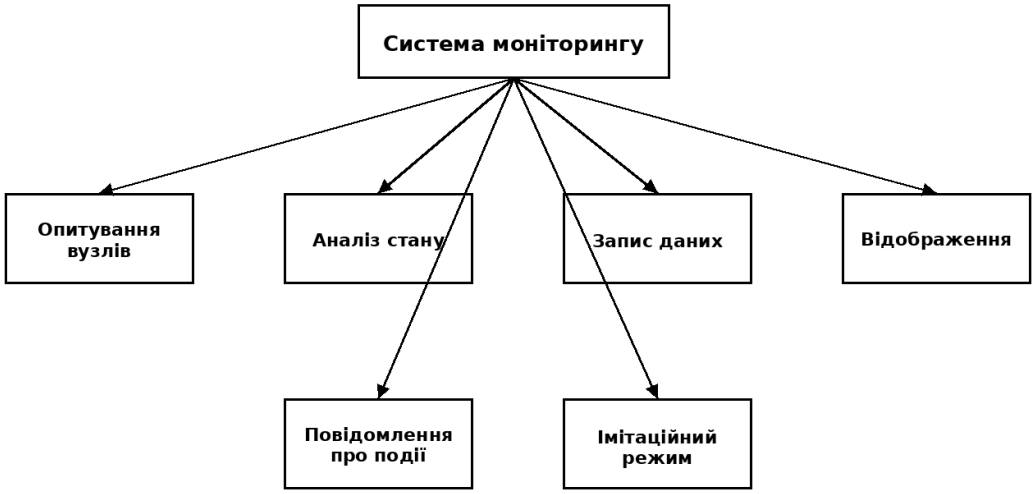


Рисунок 2.3 – Розподіл функціональних завдань між компонентами

2.2 Розподіл функцій між компонентами програмно-апаратного комплексу

Після визначення загальної архітектури потрібно описати, які завдання здатна виконувати система і які з них можуть бути розподілені між окремими компонентами. Для теми SNMP-моніторингу немає потреби розглядати всі можливі функції промислових систем, оскільки така робота стала б надмірно складною для бакалаврського рівня. Доцільніше виділити базові задачі, без яких програмно-апаратний комплекс не матиме практичного змісту. До таких задач належать опитування вузлів, перевірка доступності обладнання, отримання службових метрик, обробка результатів, збереження даних, формування журналу подій і підготовка інформації для вебінтерфейсу адміністратора.

Перше завдання опитування вузлів. Воно полягає у тому, що система бере зі списку IP-адресу пристрою, версію SNMP, параметри доступу і набір OID, після чого формує запит до відповідного SNMP-агента. У реальній системі такий список може зберігатися у базі даних або окремому конфігураційному файлі. Для демонстраційного прототипу достатньо використати підготовлений перелік вузлів, який імітує роботу з реальним обладнанням. Якщо вузлів декілька, опитування кожного з них можна виконувати незалежно. Наприклад, опитування маршрутизатора не потребує очікування відповіді від джерела безперебійного живлення, а перевірка сервера не залежить від стану комутатора. Саме тому ця задача добре підходить для розпаралелювання або хоча б для логічного розділення на окремі підзадачі.

Друге завдання визначення доступності вузла. На перший погляд воно є простим, оскільки потрібно лише з’ясувати, чи відповідає пристрій на запит. Однак на практиці ця задача має кілька нюансів. Вузол може не відповісти один раз через короткочасну затримку в мережі, перевантаження пристрою або тимчасову проблему з каналом зв’язку. Один timeout не завжди означає повну відмову обладнання. Тому архітектура повинна дозволяти фіксувати факт

відсутності відповіді, зберігати час останньої успішної перевірки та змінювати статус вузла не випадково, а за зрозумілим правилом. У демонстраційному прототипі достатньо використовувати стани Online і Offline, але в подальшому систему можна розширити станами Warning, Unknown або Maintenance.

Третє завдання збір службових метрик. Для мінімального прототипу достатньо використовувати такі параметри, як sysName і sysUpTime, оскільки вони добре демонструють факт отримання SNMP-відповіді від пристрою. Значення sysName дозволяє ідентифікувати вузол, а sysUpTime показує час безперервної роботи обладнання після останнього перезавантаження. У розширеній версії до цих параметрів можна додати стан мережевих інтерфейсів, лічильники вхідного та вихідного трафіку, кількість помилок, показники завантаження або параметри живлення. Важливо, щоб система не була жорстко прив’язана до однієї конкретної метрики. Краще, щоб перелік параметрів задавався окремо, оскільки тоді комплекс буде простіше пристосувати до іншого обладнання.

Четверте завдання обробка отриманих результатів. Після отримання відповіді система повинна не лише прийняти значення, а й визначити, що саме воно означає для стану вузла. Якщо SNMP-відповідь отримано, вузол може залишатися у стані Online. Якщо відповідь відсутня або виникає timeout, система повинна сформувати повідомлення про помилку. Якщо значення метрики виходить за допустимі межі, у майбутньому може створюватися попередження. У демонстраційному варіанті така логіка може бути спрощеною, але її все одно потрібно описати, оскільки саме вона перетворює набір окремих значень у корисну інформацію для адміністратора.

П’яте завдання збереження даних. Після отримання відповіді значення потрібно не лише показати на екрані, а й записати у базу даних. Це дає можливість переглядати історію, будувати графіки і порівнювати стан вузла в різні моменти часу. У спрощеній реалізації достатньо зберігати дату, ідентифікатор вузла, назву метрики та її значення. Така інформація вже дозволяє

пояснити логіку історичного моніторингу. Наприклад, адміністратор може побачити, коли востаннє вузол був доступний, які значення були отримані та чи змінювався стан пристрою протягом певного часу.

Шосте завдання формування журналу подій. Система не повинна записувати кожну дрібну дію як окрему важливу подію, оскільки це зробить журнал перевантаженим і незручним для перегляду. До журналу доцільно вносити ті зміни, які мають значення для адміністратора: успішне опитування, помилку доступу, timeout, перехід вузла у стан Offline або повернення у стан Online. Журнал подій робить прототип більш схожим на реальну систему моніторингу, тому що адміністратор працює не тільки з поточними значеннями, а й з історією інцидентів. Саме журнал дозволяє швидко зрозуміти, коли виникла проблема і як довго вона тривала.

Сьоме завдання підготовка даних для вебінтерфейсу. Воно відрізняється від звичайного виведення тексту, оскільки для панелі моніторингу потрібно перетворити сирі записи бази у зрозумілий вигляд. Наприклад, останній запис про вузол може перетворюватися на поточний статус, кількість подій за певний період на показник у картці, а історичні значення на графік. Ця задача може виконуватися окремо від SNMP-опитування, тому вебінтерфейс не повинен чекати кожного мережевого запиту. Він має працювати з уже підготовленою інформацією, яка зберігається у базі даних.

Окремо варто зазначити, що частину завдань можна виконувати паралельно або незалежно одне від одного. Наприклад, опитування різних вузлів може бути поділене на окремі задачі, оскільки результат одного пристрою не повинен блокувати перевірку іншого. Формування журналу подій також може виконуватися окремо від відображення даних у вебінтерфейсі. Збереження метрик і підготовка графіків можуть бути розділені на різні логічні етапи. Такий підхід робить систему більш гнучкою і дозволяє у майбутньому розширити її без повної перебудови архітектури.

Для цієї роботи обрано набір задач, який не потребує складної реалізації, але показує основну ідею розподіленої системи. Частина задач працює з мережею, частина з базою даних, частина з журналом подій, а частина з відображенням результатів у вебінтерфейсі. Якщо ці частини розділити, система стає зрозумілішою і легше описується у пояснювальній записці. У такому вигляді програмно-апаратний комплекс можна розглядати не як одну монолітну програму, а як набір пов'язаних компонентів, кожен із яких має власне призначення.

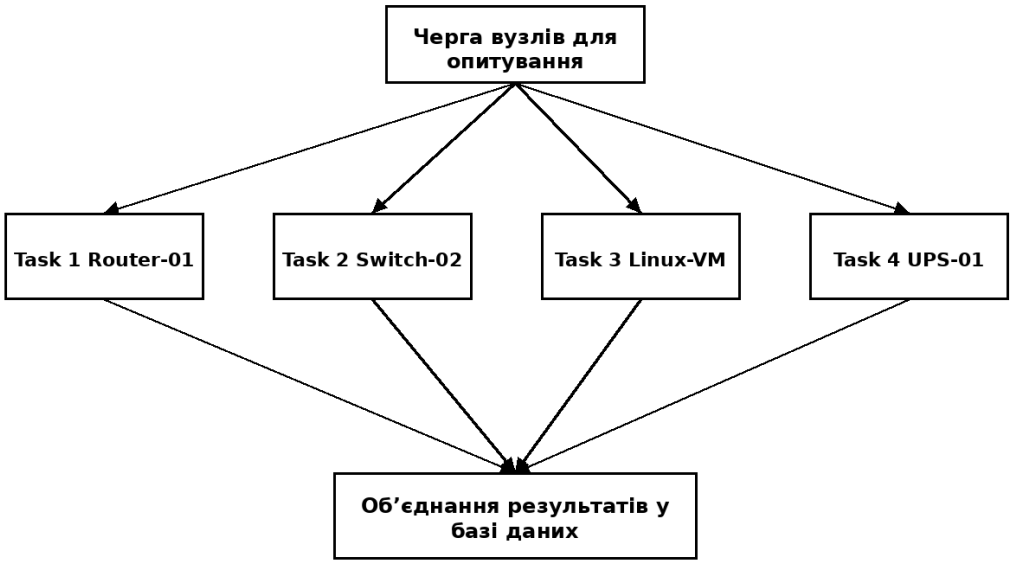


Рисунок 2.4 – Паралельне виконання задач опитування вузлів

Розпаралелювання у цій архітектурі не обов'язково означає складні обчислення або використання окремого кластера. У контексті SNMP-моніторингу достатньо того, що опитування різних вузлів може виконуватися незалежними задачами. Це зменшує час очікування і не дозволяє одному повільному пристрою зупинити перевірку всієї мережі. Наприклад, якщо UPS не відповідає, система все одно може отримати дані від маршрутизатора і сервера.

Найпростішим варіантом є послідовне опитування, коли система перевіряє вузли один за одним. Такий підхід легкий для реалізації, але має недолік:

загальний час циклу залежить від найповільніших відповідей. Якщо для одного вузла timeout становить кілька секунд, а таких вузлів багато, затримка накопичується. Тому на рівні архітектури корисно передбачити можливість незалежного виконання запитів.

Для бакалаврської роботи не потрібно реалізовувати складний планувальник задач. Достатньо описати, що вузли можуть бути поділені на групи або передані до черги опитування. Кожна задача бере вузол, виконує перевірку, формує результат і повертає його до спільного сховища. Саме така логіка показана на рисунку 2.4. Вона добре пояснює, чому систему можна назвати розподіленою на рівні функцій.

Окремо можна розподілити обробку подій. Після отримання метрики система не обов’язково повинна одразу блокувати весь цикл, щоб записати довге повідомлення або підготувати графік. Дані можна спочатку зберегти, а формування події виконати другим кроком. У малому прототипі це не дає великого виграшу, але в архітектурі показує правильний напрям розвитку.

Функція зберігання також може бути організована по-різному. У найпростішому випадку collector напряму записує дані в SQLite. У більш розвиненій системі між collector і базою може бути проміжний буфер або черга. Це потрібно для того, щоб короткі збої бази або велика кількість відповідей не призводили до втрати даних. У межах цієї роботи достатньо описати таку можливість як напрям подальшого розвитку.

Вебінтерфейс, на відміну від collector, працює з уже підготовленими даними. Це дає змогу відкривати панель моніторингу незалежно від того, чи виконується зараз черговий цикл опитування. Якщо цикл ще не завершився, користувач бачить останні доступні значення. Такий підхід зручний, тому що адміністратор не чекає завершення мережевих запитів при кожному відкритті сторінки.

Ще одна задача, яку варто виділити, - перевірка коректності отриманих значень. Наприклад, sysUpTime має бути службовим значенням часу, а не

довільним текстом. Якщо система отримує порожнє значення або помилку, вона повинна зберегти не тільки сам факт помилки, а й пояснення. У демонстраційному режимі це можна показати через події типу Warning або Timeout.

Таким чином, функціонал системи можна представити як набір взаємопов’язаних, але відносно незалежних задач. Одні задачі відповідають за мережеву частину, інші - за збереження, ще інші - за інтерфейс і події. Це дає можливість описати комплекс не як одну монолітну програму, а як систему компонентів, які виконують свої ролі.

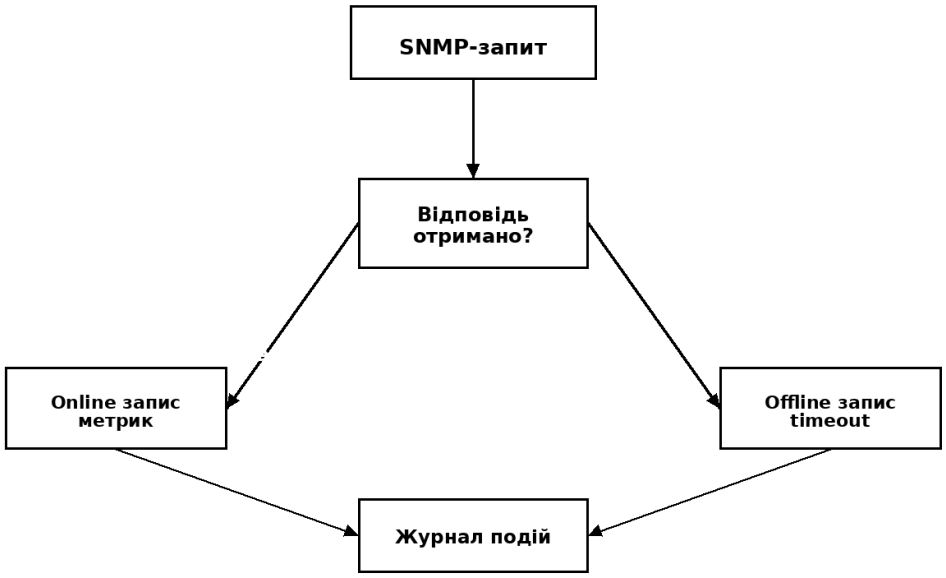


Рисунок 2.5 – Формування стану вузла після виконання опитування

Для опису програмно-апаратного комплексу важливо не тільки перелічити функції, а й показати, які дані переходять між компонентами. Спочатку система отримує список вузлів. Далі для кожного вузла формується завдання опитування. Після виконання запиту результат перетворюється на поточний стан, метрику або подію. На останньому етапі підготовлені дані відображаються у вебінтерфейсі.

У цьому ланцюжку є кілька місць, де дані можуть оброблятися паралельно. По-перше, можна паралельно перевіряти кілька вузлів. По-друге, можна окремо формувати журнал подій. По-третє, вебінтерфейс може працювати незалежно від активного циклу опитування. Таке розділення не робить прототип складним, але дає змогу пояснити архітектурну логіку більш переконливо.

Для демонстраційного прототипу обрано мінімальний набір результатів: status, sysName, sysUpTime і події. Проте сама структура дозволяє додати й інші метрики. Наприклад, для комутатора можна додати стан портів, для маршрутизатора - обсяг трафіку, для UPS - стан батареї. Це означає, що система не обмежується одним типом обладнання, а може розширюватися за рахунок нових OID і профілів вузлів.

При проєктуванні функціоналу також враховується, що адміністратор не повинен вручну запускати кожну дрібну операцію. Його роль полягає у налаштуванні вузлів і перегляді результатів. Внутрішні завдання - опитування, запис, перевірка стану, створення події - повинні виконуватися системою самостійно. Саме це підводить до поняття самоорганізації, розглянутого у наступному підрозділі.

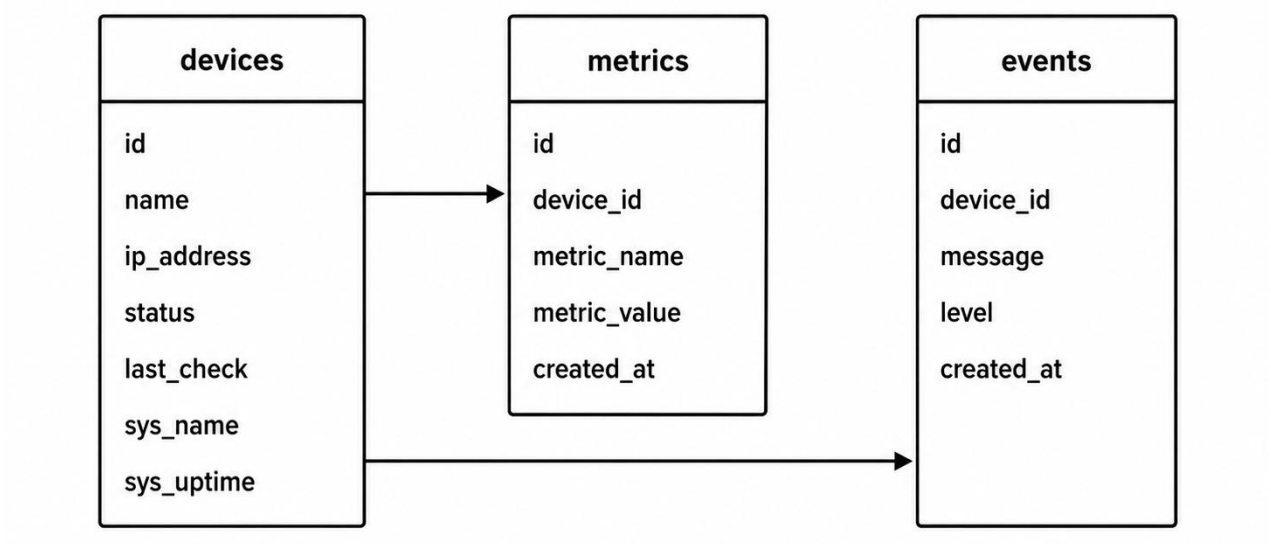


Рисунок 2.6 – Логічна структура бази даних у складі системи

2.3 Вибір принципу автономної роботи програмно-апаратного комплексу моніторингу

Самоорганізацію в межах цієї кваліфікаційної роботи не потрібно розуміти як складний штучний інтелект або повністю автономну систему. Для простого комплексу моніторингу достатньо описати здатність системи самостійно виконувати повторювані дії, змінювати стан вузлів, створювати події і частково перебудовувати логіку роботи залежно від результатів опитування. Такий підхід реалістичний для бакалаврської роботи і добре узгоджується з темою SNMP-моніторингу.

Основним проявом самоорганізації є автоматичне виконання циклу контролю. Адміністратор задає перелік вузлів і параметри доступу, а система самостійно проходить по цьому переліку, виконує запити, обробляє відповіді та оновлює стан. Навіть якщо у прототипі запуск виконується вручну через кнопку, архітектура має бути описана так, щоб надалі цей процес можна було замінити на періодичний планувальник.

Другий прояв самоорганізації - автоматичне визначення стану вузла. Система не чекає, поки адміністратор сам вирішить, чи працює пристрій. Вона порівнює результат опитування з очікуваним: якщо відповідь є, вузол отримує стан Online; якщо відповіді немає, створюється подія timeout і вузол може отримати стан Offline. Це проста логіка, але саме вона робить моніторинг корисним.

Третій прояв - зміна поведінки після помилки. Наприклад, якщо вузол не відповів, система може записати подію і під час наступного циклу перевірити його ще раз. Якщо помилка повторюється, вузол залишається у стані Offline. Якщо відповідь знову з’являється, стан змінюється на Online, а в журналі фіксується відновлення. Так система сама підтримує актуальну картину стану мережі.

Для демонстраційної роботи можна описати також спрощений механізм пріоритетів. Вузли, які перебувають у стані Warning або Offline, можуть перевірятися частіше, ніж вузли у нормальному стані. Це не обов’язково реалізовувати в повному коді, але як архітектурне рішення воно показує, що система може адаптуватися до ситуації. Такі правила є простими і зрозумілими, тому не виглядають надуманими.

Рисунок 2.7 показує логіку самоорганізації на рівні подій. Після отримання інформації система вибирає режим обробки: нормальний або аварійний. У нормальному режимі достатньо оновити значення. В аварійному режимі потрібно сформувати подію, змінити статус і показати це у вебінтерфейсі. Така схема пояснює, як система реагує без прямої участі адміністратора.

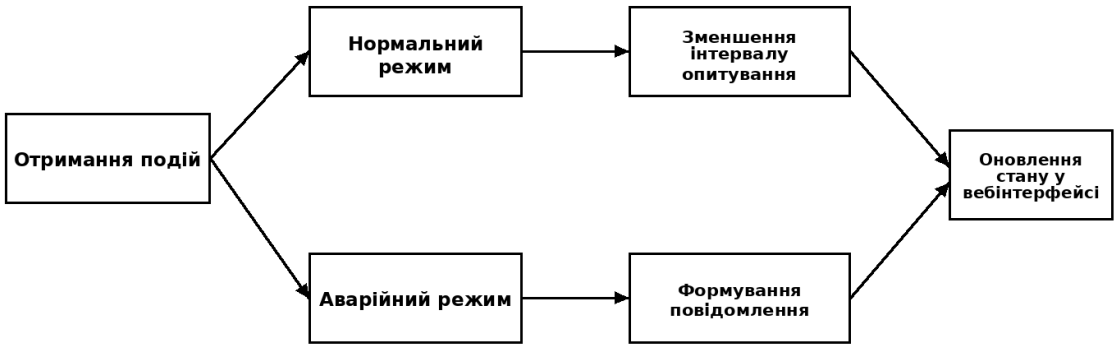


Рисунок 2.7 – Логіка самоорганізації комплексу моніторингу

Окремим питанням є перебудова архітектури. У промислових системах це може означати автоматичне додавання нових вузлів, зміну маршрутів, балансування навантаження або перемикання між серверами. У межах цієї роботи достатньо розглянути простішу модель: якщо основний collector недоступний або перевантажений, його роль може бути передана іншому компоненту. Це не обов’язково фізично реалізовувати у прототипі, але такий механізм можна закласти на рівні архітектури.

Переміщення центру між компонентами означає, що система не повинна жорстко залежати від одного процесу. Наприклад, база даних зберігає список вузлів і останній стан, тому інший collector може продовжити роботу, прочитавши ці дані. Вебінтерфейс також не залежить від конкретного collector, тому що працює з базою. Завдяки цьому центр виконання задач може змінюватися без зміни способу перегляду результатів адміністратором.

У найпростішому варіанті такий підхід можна описати як резервний режим. Основний модуль виконує опитування, а резервний може бути підключений у разі помилки. Для дипломної роботи це достатньо показати схемою і поясненням. Головне - не заявляти, що створено повністю відмовостійкий кластер, якщо фактично йдеться про демонстраційний прототип. Правильніше писати, що архітектура допускає подальше розширення у цьому напрямі.

Ще один варіант самоорганізації пов’язаний із тестовим режимом. Якщо реальні SNMP-відповіді недоступні, система може працювати з демонстраційним набором даних. Це дозволяє не зупиняти перевірку вебінтерфейсу і журналу подій. З погляду архітектури тестовий режим є допоміжним джерелом даних, яке може замінити реальні вузли під час навчальної демонстрації.

Для адміністратора така організація має практичну користь. Він не повинен вручну оновлювати статуси, переписувати журнал або перевіряти кожен вузол окремо. Система сама збирає інформацію, сама обробляє помилки і сама показує результат. Адміністратор залишається відповідальним за інтерпретацію результатів і прийняття рішень, але рутинна частина моніторингу автоматизується.

Схема переміщення центру між компонентами показана на рисунку 2.8. Вона демонструє, що collector не є єдиною точкою взаємодії з користувачем. База даних і журнал залишаються спільним шаром, через який інші частини системи можуть отримувати актуальну інформацію.

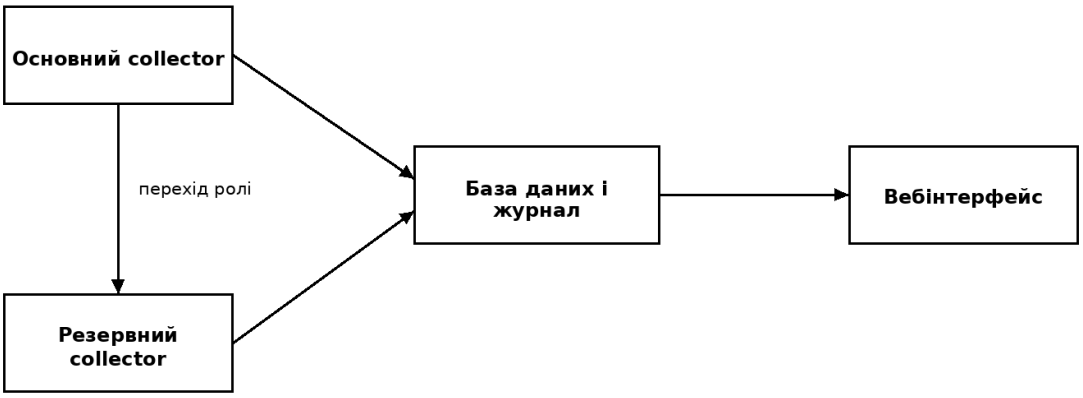


Рисунок 2.8 – Переміщення центру виконання задач між компонентами

Самоорганізація також проявляється у підготовці даних для користувача. Система може сама визначити, які значення потрібно показати на головній панелі: кількість вузлів, кількість Online, кількість Offline і кількість подій. Для цього не потрібно, щоб адміністратор вручну рахував записи у базі. Вебінтерфейс отримує вже узагальнену інформацію і подає її у вигляді простих показників.

При цьому вебінтерфейс не повинен перетворюватися на окремий складний центр керування. У даній роботі його роль обмежується переглядом стану і результатів. Такий підхід відповідає принципу read-only моніторингу. Зміна налаштувань мережевого обладнання через вебпанель не передбачається, тому ризик випадкового впливу на реальну інфраструктуру зменшується.

Місце вебінтерфейсу в архітектурі показано на рисунку 2.9. Він знаходиться після бази даних і журналу подій, тобто працює з обробленою інформацією. Це важливо, тому що в іншому випадку кожне відкриття сторінки могло б запускати мережеві запити, збільшувати затримки і створювати зайве навантаження на пристрої.

Узагальнюючи, можна сказати, що самоорганізація для даного комплексу складається з кількох простих механізмів: автоматичне проходження переліку вузлів, визначення стану за результатом відповіді, створення подій, можливість повторного опитування, підтримка тестового режиму і потенційне перенесення

задач між collector-компонентами. Ці механізми не ускладнюють прототип, але роблять його архітектуру логічною і придатною для пояснення у дипломній роботі.

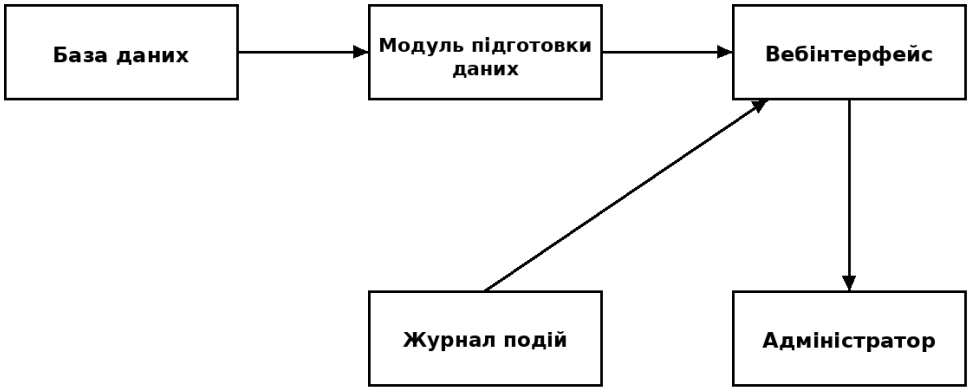


Рисунок 2.9 – Місце вебінтерфейсу в архітектурі системи

Для запропонованого підходу важливо не перебільшувати рівень автоматизації. Система не приймає складних управлінських рішень і не змінює конфігурацію обладнання. Вона виконує іншу, більш обмежену, але корисну задачу: збирає інформацію, упорядковує її та показує стан вузлів. Саме така обережна постановка задачі відповідає реальному рівню демонстраційного прототипу.

У перспективі описану архітектуру можна доповнити автоматичним виявленням вузлів, підтримкою SNMPv3 у повному обсязі, механізмом сповіщень, резервним collector і розмежуванням прав користувачів. Однак у межах бакалаврської роботи достатньо показати базовий принцип. Якщо одразу додати занадто багато можливостей, робота буде виглядати менш цілісною, а пояснення стане складнішим.

Таким чином, третій підрозділ другого розділу показує, що навіть проста система SNMP-моніторингу може мати елементи самоорганізації. Вона не потребує від адміністратора постійних ручних перевірок, може сама реагувати на відсутність відповіді, вести журнал подій і підтримувати актуальний стан у

вебінтерфейсі. Для теми кваліфікаційної роботи цього достатньо, щоб перейти до опису реалізації демонстраційного прототипу у третьому розділі.

Для того щоб архітектура не залишалась лише теоретичною схемою, її потрібно прив’язати до простого прикладу роботи. У демонстраційному варіанті можна вважати, що система контролює чотири умовні вузли: маршрутизатор Router-01, комутатор Switch-02, сервер Linux-VM та джерело безперебійного живлення UPS-01. Такий набір не є випадковим. Він показує різні типи обладнання, які зазвичай можуть бути присутні у невеликому телекомунікаційному вузлі або серверній шафі.

Маршрутизатор у такій схемі відповідає за зв’язок вузла з іншими сегментами мережі. Для нього важливо знати не тільки факт доступності, а й час роботи, стан інтерфейсів і наявність помилок. Якщо маршрутизатор недоступний, це може означати, що весь віддалений вузол фактично відрізаний від основної мережі. Тому в системі моніторингу такий пристрій має один із найвищих пріоритетів для перевірки.

Комутатор виконує іншу роль. Він може залишатися доступним, але окремі порти можуть бути вимкнені або мати помилки. У мінімальному прототипі це не обов’язково деталізувати, однак на рівні архітектури потрібно передбачити, що один вузол може мати кілька контрольованих параметрів. Це ще раз підтверджує необхідність таблиці метрик, а не одного поля зі станом пристрою.

Сервер Linux-VM у прикладі використовується як зручний навчальний вузол. На ньому можна уявно розгорнути SNMP-агент або замінити його відповідь тестовими даними. Такий вузол корисний для пояснення, тому що серверні операційні системи зазвичай дають більше службової інформації, ніж прості мережеві пристрої. У майбутньому до нього можна було б додати контроль навантаження, пам’яті або дисків.

Джерело безперебійного живлення показує інший тип контрольованого об’єкта. Воно може не бути основним мережевим пристроєм, але його стан напряму впливає на стабільність вузла. Якщо UPS переходить у стан Warning або

Offline, адміністратор має це побачити раніше, ніж відбудеться повне вимкнення обладнання. Тому в архітектурі передбачено не тільки метрики мережі, а й події, пов’язані з живленням.

На практиці під час моніторингу часто виникає ситуація, коли частина вузлів відповідає, а частина ні. Якщо система побудована як єдиний послідовний сценарій без обробки помилок, один недоступний вузол може зіпсувати весь цикл перевірки. У запропонованій архітектурі такого бути не повинно: помилка одного пристрою перетворюється на подію, але інші вузли продовжують оброблятися далі.

Саме через це у другому розділі зроблено акцент на розподілі задач. Для невеликої роботи це може здаватися зайвим, але така логіка допомагає уникнути типових помилок. Наприклад, не варто змішувати в одному місці отримання SNMP-відповіді, запис у базу, побудову графіка і відображення сторінки. Коли ці ролі розділені, легше пояснити, де саме виникла помилка і що потрібно змінити.

Ще один практичний момент стосується оновлення даних. У реальній системі адміністратор не повинен щоразу вручну перезавантажувати сторінку або виконувати команду. Достатньо, щоб система періодично оновлювала записи, а вебінтерфейс показував останній стан. У демонстраційному прототипі можна залишити ручний запуск, але в архітектурі доцільно одразу описати можливість автоматичного циклу.

Під час проєктування також потрібно врахувати, що не всі значення мають однакову важливість. Наприклад, sysName потрібний переважно для ідентифікації пристрою, а статус Online/Offline безпосередньо впливає на реакцію адміністратора. Час роботи sysUpTime допомагає зрозуміти, чи не перезавантажувався вузол, але сам по собі не завжди є аварійним параметром. Через це система повинна зберігати метрики, але окремо виділяти події.

Події в архітектурі можна розглядати як короткі повідомлення про важливі зміни. Якщо вузол був Online і став Offline, це подія. Якщо вузол не відповів

через timeout, це також подія. Якщо після помилки він знову почав відповідати, це подія відновлення. Така логіка проста, але вона робить прототип більш схожим на реальний інструмент адміністратора, а не просто на таблицю з тестовими числами.

Важливо, щоб самоорганізація системи не виглядала відірваною від теми. У цій роботі вона не означає, що система сама ремонтує мережу або змінює конфігурацію маршрутизаторів. Мова йде про більш реалістичні речі: автоматичний вибір дії після відповіді або помилки, запис події, оновлення статусу і підготовку інформації для користувача. Для бакалаврської роботи цього рівня достатньо.

Якщо розглядати архітектуру з погляду подальшого розвитку, то найпростішим напрямом є додавання профілів пристроїв. Наприклад, для маршрутизатора можна мати один набір OID, для комутатора - інший, для UPS - третій. Тоді система зможе працювати з різним обладнанням без повного переписування логіки. У демонстраційному прототипі це можна описати як можливість розширення, навіть якщо фактично використано лише базові параметри.

Другим напрямом розвитку є додавання рівнів важливості подій. Не кожна подія потребує однакової реакції. Успішне опитування можна вважати інформаційним повідомленням, timeout - попередженням, а довгу відсутність відповіді - критичною подією. Такий поділ допомагає адміністратору не губитися у великій кількості записів журналу.

Третім напрямом є розділення ролей користувачів. У простому прототипі всі дані може переглядати один адміністратор. У більшій системі можуть бути оператори, які лише переглядають стан, і адміністратори, які змінюють налаштування вузлів. Для цієї роботи реалізація авторизації не є обов’язковою, але сама архітектура не повинна заважати її додаванню в майбутньому.

Запропонована архітектура також добре підходить для пояснення графічної частини. Перше креслення може показувати загальну структурну

схему комплексу, друге - структуру зберігання даних, третє - алгоритм опитування. Такий набір пов’язаний із текстом розділу і не виглядає випадковим. Він показує систему з трьох боків: склад компонентів, дані та послідовність дій.

Таким чином, другий розділ не дублює третій. Тут не потрібно показувати готові скріншоти або доводити, що прототип уже працює. Завдання цього розділу інше: пояснити, чому система має саме таку структуру, які компоненти потрібні, які задачі між ними розподіляються і як комплекс може працювати майже самостійно після початкового налаштування.

Описаний підхід є достатньо простим для виконання, але водночас не зводиться до однієї статичної картинки. У ньому є джерела даних, обробка, зберігання, події, інтерфейс і резервні сценарії. Саме така структура дозволяє далі перейти до третього розділу, де архітектурні рішення подаються вже у вигляді демонстраційного прототипу з тестовими SNMP-даними.

2.4 Проєктування інформаційних потоків і структури даних у програмно-апаратному комплексі

У цьому підрозділі розглядається порядок проходження даних у системі моніторингу від моменту отримання інформації від телекомунікаційного вузла до її відображення у веб-базованому інтерфейсі адміністратора.

Основними інформаційними потоками у комплексі є:

1) дані про контрольовані вузли;

2) результати SNMP-опитування;

3) службові метрики;

4) події та повідомлення про помилки;

5) узагальнені дані для вебінтерфейсу.

Дані про вузли зберігаються у базі даних і містять назву пристрою, IP-адресу, версію SNMP, поточний статус і час останньої перевірки. Під час запуску опитування система зчитує цей перелік, формує запити до вузлів або отримує тестові значення у демонстраційному режимі. Після цього результати передаються до модуля оброблення.

Якщо відповідь від вузла отримано, система оновлює його стан, записує отримані метрики та фіксує час перевірки. Якщо відповідь відсутня, створюється подія типу timeout, а вузол може отримати стан Offline. Завдяки цьому адміністратор бачить не лише поточний стан пристрою, а й причину зміни цього стану.

Для збереження даних доцільно використати три основні логічні сутності: devices, metrics та events. Таблиця devices відповідає за перелік вузлів, таблиця metrics за історію отриманих значень, а таблиця events за журнал подій. Такий поділ дозволяє не змішувати опис пристроїв, числові або службові значення та повідомлення про події.

Інформаційний потік у системі можна подати у вигляді послідовності: телекомунікаційний вузол - SNMP collector - модуль оброблення - база даних - вебінтерфейс адміністратора. У демонстраційному режимі реальний вузол може бути замінений тестовим набором SNMP-даних, але загальна логіка проходження інформації залишається такою самою.

Запропонована структура даних є достатньою для демонстраційного прототипу і водночас може бути розширена у майбутньому. Наприклад, до системи можна додати таблиці користувачів, ролей доступу, шаблонів OID, порогових значень або налаштувань сповіщень. Це дозволяє розглядати комплекс не тільки як навчальний приклад, а і як основу для подальшого розвитку системи моніторингу.

2.5 Висновок до розділу 2

У другому розділі виконано проєктування архітектури розподіленої універсальної системи моніторингу віддалених телекомунікаційних вузлів. Система розглянута як набір взаємопов’язаних компонентів: віддалених вузлів, SNMP collector, модуля тестових даних, бази даних, журналу подій і вебінтерфейсу адміністратора.

Пояснено загальну архітектуру комплексу та призначення його основних частин. Показано, що така архітектура дозволяє відокремити отримання даних від їх зберігання і відображення. Це робить систему більш зрозумілою, а також дає можливість у майбутньому розширювати її без повної перебудови.

Визначено основні завдання, які можуть виконуватися компонентами системи: опитування вузлів, перевірка доступності, отримання службових SNMP-метрик, збереження історії, формування журналу подій і підготовка даних для вебінтерфейсу. Окремо обґрунтовано, що частина цих задач може виконуватися незалежно або паралельно, оскільки результати опитування різних вузлів не залежать один від одного.

Розглянуто механізми самоорганізації у межах спрощеного прототипу. До них віднесено автоматичне визначення стану вузла, створення подій, повторне опитування, підтримку тестового режиму та можливість перенесення ролі collector між компонентами у разі подальшого розвитку системи. Запропонована архітектура не є промисловою відмовостійкою платформою, але достатньо повно демонструє принципи побудови програмно-апаратного комплексу моніторингу.

Результати цього розділу є основою для третього розділу, у якому описується демонстраційний прототип системи, структура його даних, вебінтерфейс та перевірка роботи з тестовими SNMP-даними.

У подальшому під час практичної реалізації потрібно дотримуватися саме цієї логіки: спочатку створюється перелік вузлів, потім визначається спосіб отримання даних, після цього організовується збереження результатів і тільки в кінці формується інтерфейс для перегляду. Така послідовність зменшує ризик того, що інтерфейс буде існувати окремо від реальної логіки моніторингу.

Отже, розроблена архітектура є достатньою для демонстраційного рівня бакалаврської роботи. Вона не перевантажена зайвими модулями, але має всі необхідні частини для пояснення роботи SNMP-моніторингу: джерела даних, модуль опитування, сховище, журнал, вебінтерфейс і механізми реакції на помилки.

3 АЛГОРИТМІЧНЕ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ РОЗПОДІЛЕНОЇ СИСТЕМИ МОНІТОРИНГУ

3.1 Алгоритмічна організація роботи системи та опис основних процедур у вигляді псевдокоду

Алгоритмічне забезпечення комплексу визначає порядок виконання дій під час моніторингу віддалених телекомунікаційних вузлів. У спрощеному вигляді система повинна отримати перелік вузлів, сформувати завдання опитування, виконати SNMP-запит або використати тестове значення, проаналізувати результат, зберегти дані і відобразити їх у вебінтерфейсі. Саме така послідовність використовується у демонстраційному прототипі.

Для цієї роботи важливо не стільки показати великий програмний код, скільки пояснити логіку виконання дій. Тому основні алгоритми подані у вигляді псевдокоду. Псевдокод не прив’язаний до конкретної мови програмування, але показує, які кроки повинна виконувати система і як між собою пов’язані її компоненти.

Основними алгоритмічними процедурами системи є: завантаження переліку вузлів, формування завдань опитування, виконання циклу SNMP-опитування, оброблення отриманої відповіді, запис метрик до бази даних, створення подій у журналі та оновлення даних у вебінтерфейсі адміністратора. Кожна з цих процедур виконує окрему роль у загальній логіці роботи програмно-апаратного комплексу і забезпечує послідовний рух даних від контрольованого вузла до користувача системи.

Система отримує перелік телекомунікаційних вузлів, які потрібно перевірити. Для кожного вузла можуть зберігатися назва, IP-адреса, версія SNMP, поточний статус і час останнього опитування. Після цього формується завдання опитування, у якому визначається, до якого вузла потрібно звернутися та які параметри необхідно отримати.

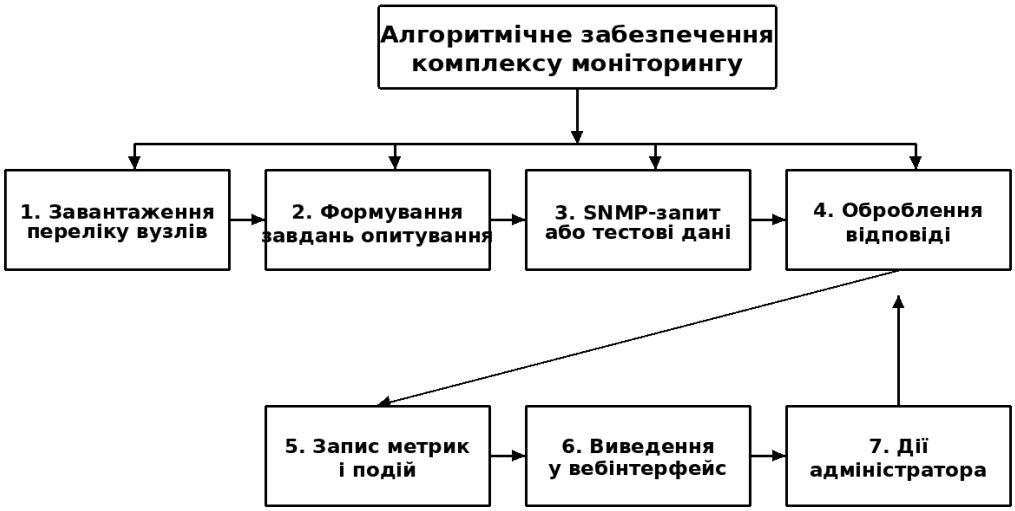


Рисунок 3.1 – Узагальнена схема алгоритмічного забезпечення системи

Як видно з рисунка 3.1, система не обмежується лише отриманням SNMP-відповіді. Після зчитування або імітації даних необхідно виконати ще кілька практичних дій: визначити стан вузла, оновити дані в сховищі, створити запис у журналі подій і передати результат до вебінтерфейсу. Це дозволяє адміністратору бачити не тільки останнє значення метрики, а й загальну картину.

Псевдокод 3.1 – Ініціалізація роботи системи

```
Початок
  завантажити список вузлів із сховища даних
  перевірити наявність активних вузлів
  якщо список вузлів порожній:
  показати повідомлення у вебінтерфейсі
  завершити підготовчий етап
 Інакше:
  сформувати чергу завдань моніторингу
  передати чергу модулю опитування
Кінець
```

Ініціалізація потрібна для того, щоб система працювала не з випадковими даними, а з визначеним переліком контрольованих вузлів. Для кожного вузла зберігаються назва, IP-адреса, версія SNMP, поточний стан та останній час перевірки. У тестовому режимі ці записи можуть бути підготовлені заздалегідь, але логіка залишається такою самою, як і в реальній системі.

Псевдокод 3.2 – Формування завдання опитування

```
Початок

 Для кожного вузла зі списку:

   створити завдання моніторингу

   додати до завдання IP-адресу вузла

   додати версію SNMP або ознаку тестового режиму

   додати перелік параметрів: sysName, sysUpTime, status

   встановити пріоритет виконання

  передати завдання в чергу опитування

Кінець
```

Формування завдання дає змогу відокремити опис вузла від самого процесу опитування. Це важливо для розподіленої системи, оскільки опис пристрою зберігається як окрема інформація, а завдання опитування формується вже на основі цього опису. Наприклад, у базі даних може зберігатися назва вузла, IP-адреса, версія SNMP, поточний статус і перелік параметрів, які потрібно контролювати. Під час запуску перевірки система не змінює сам опис вузла, а створює окреме завдання для отримання потрібних метрик.Такий підхід робить систему гнучкішою. Надалі окремі завдання можуть виконуватися різними компонентами або в різні моменти часу.

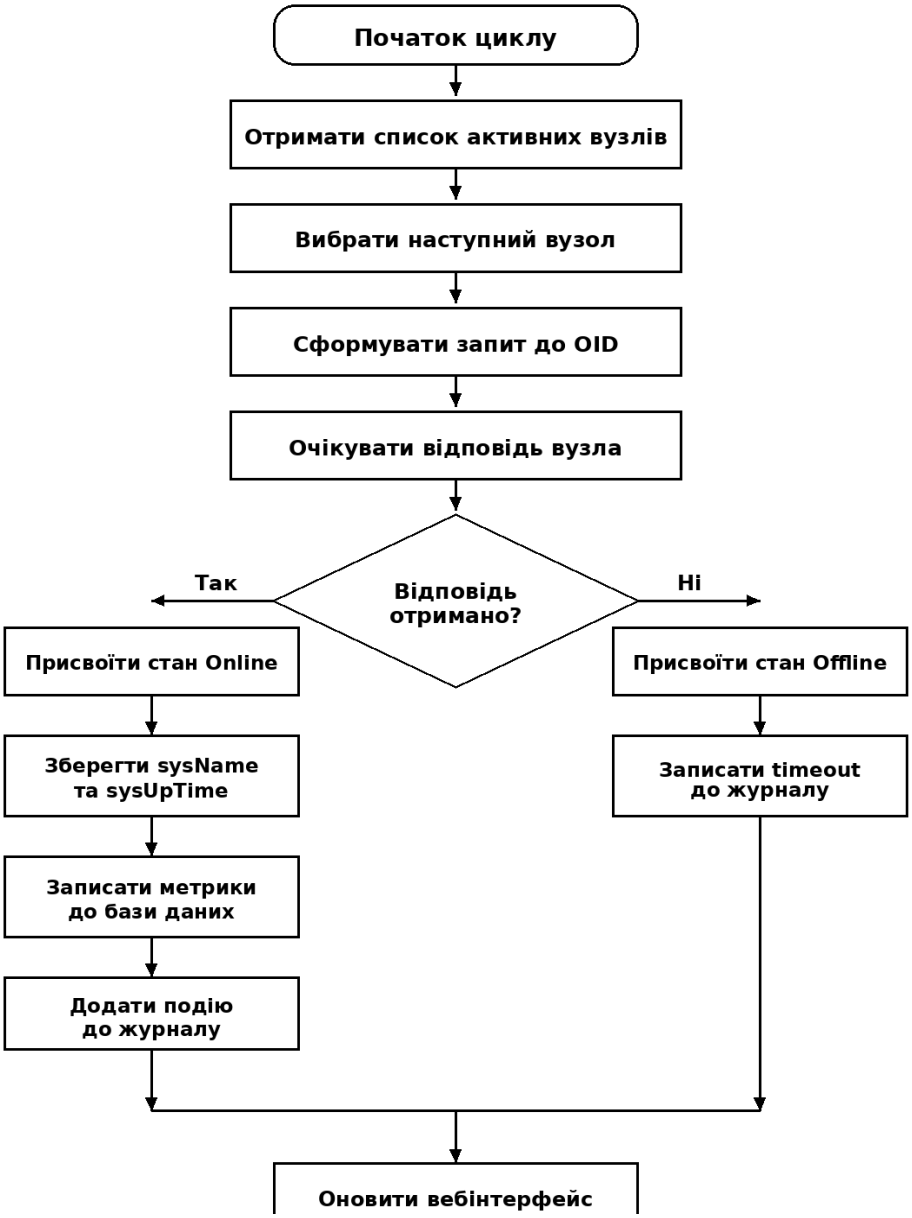


Рисунок 3.2 – Послідовність виконання циклу SNMP-опитування вузла

На рисунку 3.2 показано основний цикл опитування. Його зручно використовувати як основу для подальшого програмного коду, але сам розділ не містить конкретної реалізації мовою Python або іншою мовою. Такий спосіб подання дозволяє залишити пояснювальну записку зрозумілою навіть без перегляду додатків із кодом.

Псевдокод 3.3 – Оброблення відповіді вузла

```
Початок
Якщо відповідь від вузла отримано:
  встановити стан вузла = Online
  зчитати значення sysName
  зчитати значення sysUpTime
  сформувати запис метрики
  сформувати подію успішного опитування
Інакше:
  встановити стан вузла = Offline
  сформувати подію Timeout
  зберегти час невдалої перевірки
Кінець
```

Оброблення відповіді є ключовим етапом, оскільки саме тут відбувається перехід від технічного запиту до зрозумілого результату для адміністратора. Якщо вузол відповідає, система показує його як доступний і зберігає отримані

Псевдокод 3.4 – Створення запису в журналі подій

```
Початок
  Отримати результат перевірки вузла
  Визначити тип події: Online, Offline, Timeout або Warning
  Сформувати текст повідомлення
  Додати час створення події
  Записати подію до журналу
  Оновити блок подій у вебінтерфейсі
Кінець
```

Журнал подій потрібний для того, щоб адміністратор міг відстежити не тільки поточний стан системи, а й історію змін. Наприклад, якщо вузол зараз доступний, але протягом останньої години кілька разів переходив у стан Offline, це може вказувати на нестабільний канал зв’язку або проблему з живленням.

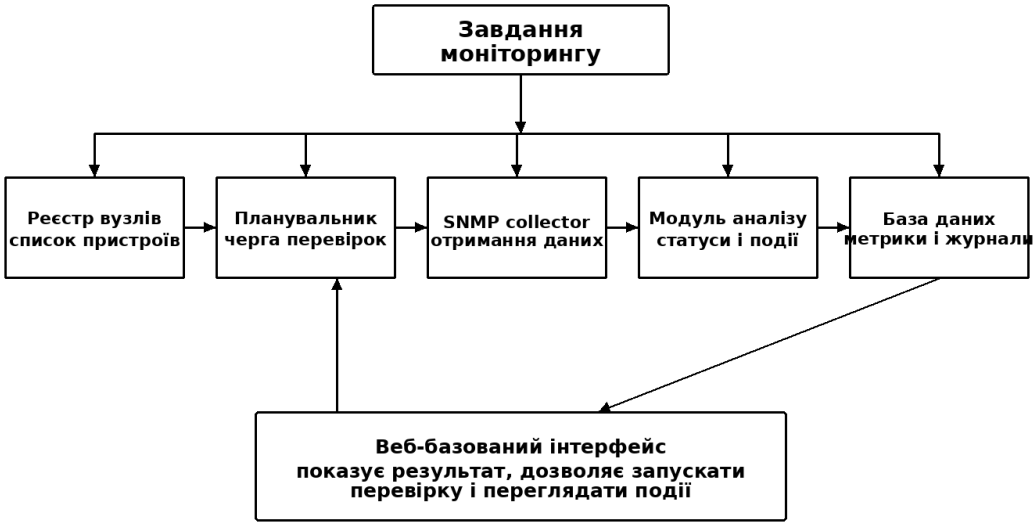


Рисунок 3.3 – Розподіл завдань між компонентами програмного забезпечення

Рисунок 3.3 показує, що у системі є не один великий блок, а кілька логічних компонентів. Така структура є зручною для пояснення розподіленої універсальної системи, тому що кожний компонент отримує свою частину роботи. Реєстр вузлів відповідає за перелік пристроїв, планувальник формує чергу перевірок, collector отримує дані, модуль аналізу визначає статус, база даних зберігає результат, а вебінтерфейс подає його користувачу.

У межах дипломної роботи достатньо розглядати обмежений набір параметрів. Для демонстрації вибрано sysName, sysUpTime, умовний стан інтерфейсу і тестові значення трафіку. Ці параметри легко пояснити у тексті, і вони добре показують ідею SNMP-моніторингу. У подальшому перелік OID можна розширити без зміни основної логіки роботи.

Окремо варто зазначити, що розподіленість у цьому випадку не означає обов'язкову наявність багатьох серверів. Навіть у простому прототипі можна виділити різні функціональні ролі: вузол як джерело даних, collector як отримувач, база даних як сховище і вебінтерфейс як засіб доступу. Такий поділ уже дозволяє показати універсальність системи.

Якщо перенести цю логіку на реальне обладнання, то основні зміни стосуватимуться лише джерела даних. Замість тестового набору значень модуль опитування буде виконувати реальні SNMP-запити до пристроїв. Інші частини системи журнал, база даних, інтерфейс і логіка станів можуть залишатися майже такими самими.

Перевага подання алгоритмів псевдокодом полягає в тому, що воно не прив'язує роботу до конкретної бібліотеки. Це корисно для пояснювальної записки, бо керівник або член комісії може зрозуміти логіку незалежно від того, чи використовується PySNMP, Net-SNMP, внутрішній API або підготовлений тестовий набір даних.

3.2 Структура та склад програмного забезпечення демонстраційної системи моніторингу

Програмне забезпечення демонстраційної системи умовно поділяється на кілька рівнів. Перший рівень відповідає за відображення інформації у вебінтерфейсі. Другий рівень виконує логіку опитування, оброблення результатів і формування подій. Третій рівень забезпечує зберігання інформації про вузли, метрики і журнал подій.

Такий поділ не є надмірно складним, але він дозволяє пояснити роботу системи як розподілену універсальну структуру. Кожен компонент має власне призначення, а взаємодія між компонентами відбувається через дані та події. У практичній реалізації це може бути набір окремих модулів або один застосунок, у якому логіка розділена на функціональні частини.

Основними складовими програмного забезпечення є модуль реєстру вузлів, модуль формування завдань, модуль SNMP-опитування, модуль аналізу відповідей, база даних, журнал подій і веб-базований інтерфейс. Для демонстраційного прототипу цього достатньо, щоб показати як внутрішню логіку, так і зовнішній вигляд системи.

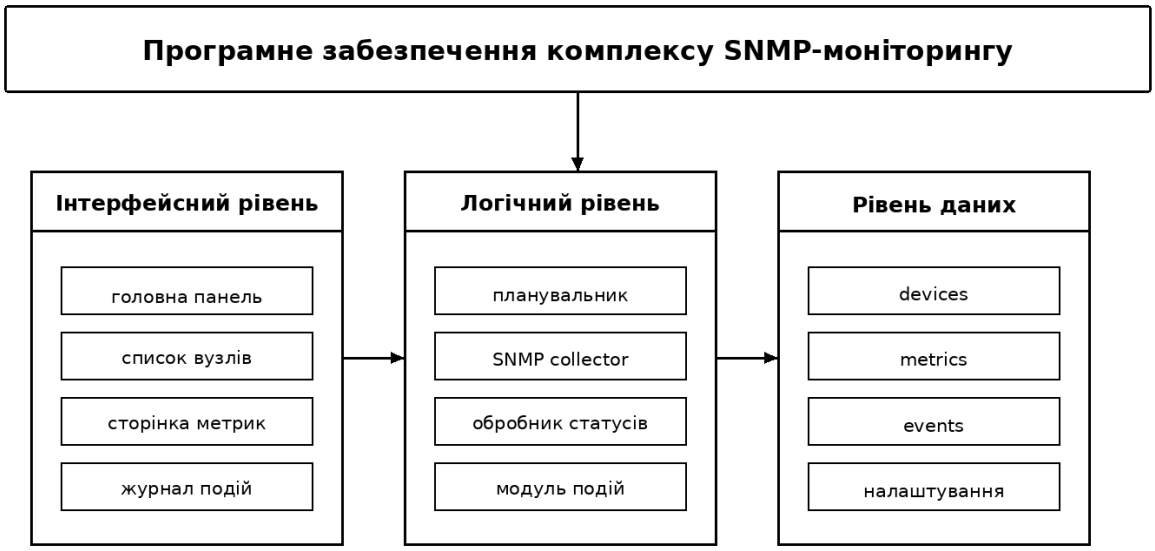


Рисунок 3.4 – Структура та склад програмного забезпечення комплексу

На рисунку 3.4 програмне забезпечення поділено на три рівні. Інтерфейсний рівень відповідає за головну панель, список вузлів, сторінку метрик і журнал подій. Логічний рівень містить планувальник, collector, обробник статусів і модуль подій. Рівень даних містить таблиці або інші структури, у яких зберігаються вузли, метрики та службові повідомлення.

Модуль реєстру вузлів зберігає базову інформацію про контрольовані пристрої. Для кожного вузла потрібні назва, IP-адреса, тип пристрою, версія SNMP, поточний статус і час останньої перевірки. У демонстраційному варіанті ці дані можуть бути задані вручну, а в майбутньому можуть вводитися через форму вебінтерфейсу.

Модуль формування завдань визначає, які вузли треба перевірити і які параметри потрібно отримати. У простому варіанті всі вузли перевіряються

однаково. У більш розвиненому варіанті можна задавати різний інтервал перевірки, різні профілі OID і різний пріоритет для важливих вузлів.

SNMP collector є компонентом, який відповідає за фактичне отримання даних. У реальному режимі він формує SNMP-запити до вузлів. У тестовому режимі він повертає підготовлені значення, які імітують відповіді. Завдяки цьому можна показати роботу всієї системи навіть без фізичного обладнання.

Модуль аналізу відповідей перетворює технічний результат опитування на зрозумілий стан. Якщо відповідь отримана, вузол позначається як Online. Якщо відповідь відсутня, система встановлює стан Offline. Якщо значення не критичне, але потребує уваги, може використовуватися стан Warning.

База даних у прототипі виконує роль спільного сховища. Вона не тільки зберігає останній стан вузлів, а й дозволяє накопичувати історію. Історичні записи потрібні для побудови графіків, аналізу стабільності вузла і підготовки звітів. Навіть проста структура з трьох сутностей дозволяє продемонструвати принцип роботи.

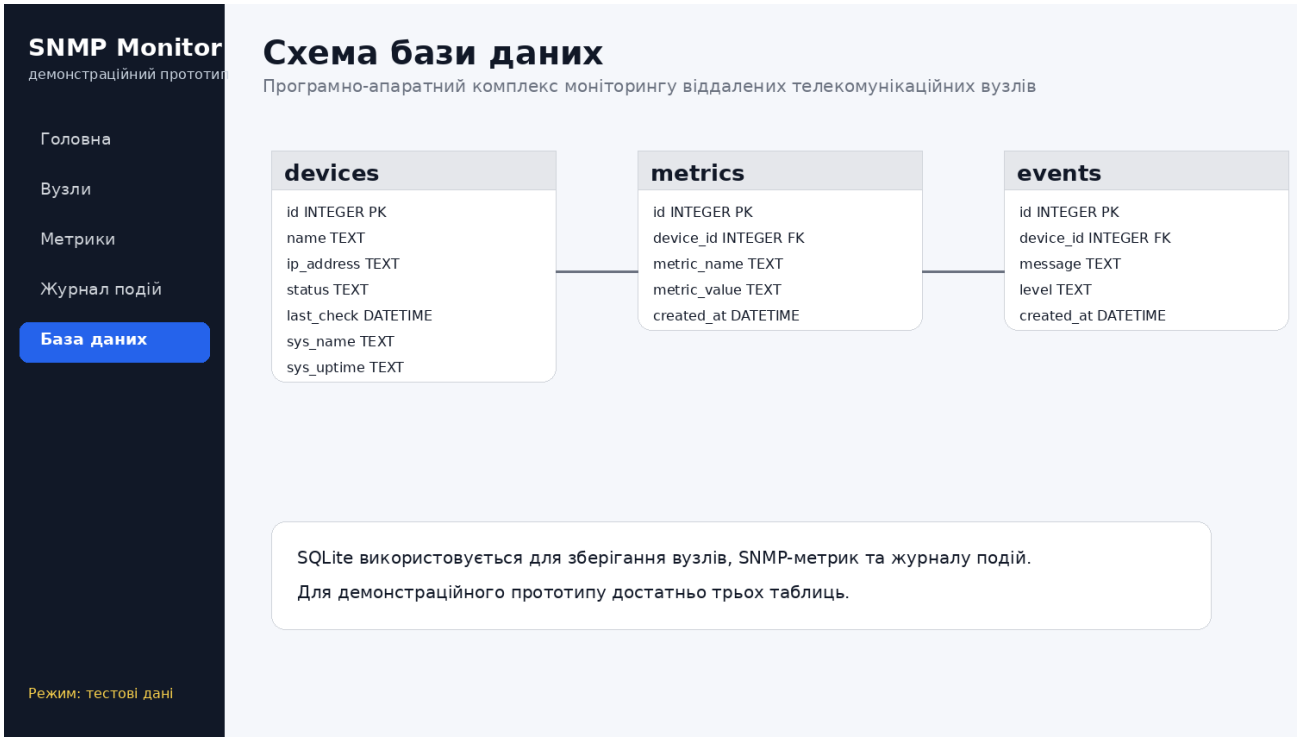


Рисунок 3.5 – Схема бази даних демонстраційного прототипу

На рисунку 3.5 показано макет структури даних, яка використовується у демонстраційному прототипі. У центрі знаходяться три основні сутності: devices, metrics та events. Таблиця devices описує контрольовані вузли, metrics зберігає значення параметрів, а events фіксує події, які виникають під час роботи системи.

У такій структурі таблиця devices є головною, бо саме від неї залежать записи метрик і подій. Якщо видалити або змінити вузол, пов’язані з ним записи повинні залишатися зрозумілими для подальшого аналізу. Тому в реальній системі для таких зв’язків доцільно використовувати ідентифікатори вузлів, а не лише їх назви.

Для демонстраційного прототипу важливо не перевантажувати структуру даних. Якщо відразу додати користувачів, ролі, складні шаблони OID і налаштування прав доступу, система стане важчою для пояснення. Тому на цьому етапі достатньо трьох основних сутностей, а розширення можна описати як напрям подальшої роботи.

Журнал подій у цій системі є практично важливою частиною, оскільки він зберігає не тільки помилки. У ньому можуть фіксуватися успішні перевірки, зміна статусу, timeout, повернення вузла у стан Online і попередження. Це дозволяє бачити послідовність подій, а не лише останній стан.

Структура програмного забезпечення також передбачає режим тестових даних. Він використовується для перевірки інтерфейсу та логіки без реального підключення до SNMP-пристроїв. У межах бакалаврської роботи це дає можливість показати результат роботи системи на зрозумілому стенді, не прив’язуючись до конкретного маршрутизатора або комутатора.

У межах бакалаврської роботи такий підхід є практичним, оскільки не завжди є можливість підключити реальний маршрутизатор, комутатор або джерело безперебійного живлення. Крім того, робота з реальним обладнанням потребує додаткового налаштування SNMP-агентів, доступу до мережі, визначення OID та дотримання вимог безпеки.

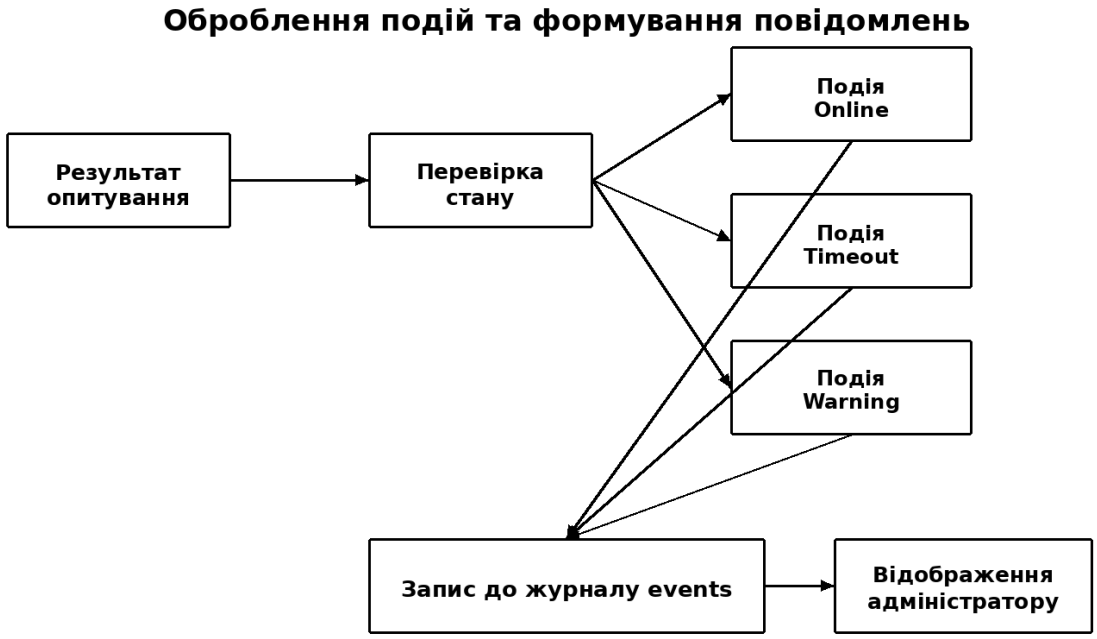


Рисунок 3.6 – Оброблення подій та формування повідомлень у системі

Рисунок 3.6 пояснює, як система працює з подіями. Після кожної перевірки результат передається до модуля аналізу. Залежно від отриманого значення формується подія Online, Timeout або Warning. Після цього подія записується у журнал і стає доступною у вебінтерфейсі.

Такий підхід зручний тим, що адміністратор не повинен самостійно аналізувати кожний технічний параметр. Система перетворює результат опитування на зрозумілий запис. Наприклад, замість перегляду сирої відповіді SNMP адміністратор бачить повідомлення про те, що вузол не відповів або успішно пройшов перевірку.

У майбутньому цей модуль можна доповнити пороговими значеннями для окремих метрик. Наприклад, якщо обсяг трафіку на інтерфейсі перевищує заданий рівень, система може створювати попередження для адміністратора. Якщо кількість помилок передавання даних різко зростає, це також може бути підставою для формування події типу Warning. У випадку, коли мережевий інтерфейс переходить у стан down, доцільно створювати аварійну подію, оскільки така ситуація може свідчити про втрату з’єднання або несправність обладнання.

3.3 Організація веб-базованого інтерфейсу для доступу до системи та постановки завдань моніторингу

Веб-базований інтерфейс є основним способом взаємодії адміністратора із системою. Його призначення полягає не лише у відображенні готових даних, а й у можливості запускати або формувати завдання моніторингу. У простому демонстраційному варіанті такими завданнями є перегляд стану вузлів, запуск перевірки, вибір конкретного вузла та аналіз журналу подій.

Перевага веб-базованого підходу полягає в тому, що адміністратору не потрібно встановлювати окрему клієнтську програму. Доступ до системи може виконуватися через браузер. Це особливо зручно для віддалених вузлів, оскільки адміністратор може переглядати стан системи з робочого місця, не підключаючись безпосередньо до кожного пристрою.

У межах дипломної роботи вебінтерфейс показано у вигляді макетів. Вони відображають основні екрани системи: головну панель, список вузлів, сторінку метрик, журнал подій, схему бази даних і результати тестування. Такі макети потрібні для того, щоб показати логіку роботи системи на практиці, навіть якщо повна програмна реалізація подається окремо в додатках.

Головна панель призначена для швидкого перегляду загального стану системи. На ній можуть відображатися кількість контрольованих вузлів, кількість вузлів у стані Online та Offline, останні події, а також коротка інформація про роботу комплексу. Такий екран потрібний для того, щоб адміністратор міг швидко оцінити ситуацію без переходу до детального перегляду кожного вузла.

Список вузлів використовується для відображення основної інформації про контрольовані пристрої. У ньому можуть бути наведені назва вузла, IP-адреса, тип пристрою, поточний статус, час останньої перевірки та короткий опис останньої події.

Рисунок 3.7 – Веб-базований доступ до системи та постановка завдань моніторингу

На рисунку 3.7 показано загальний принцип роботи адміністратора з вебінтерфейсом. Користувач відкриває панель моніторингу через браузер, переглядає стан вузлів, може ініціювати перевірку і отримує результат у вигляді таблиць, карток, графіків або записів журналу. Такий підхід наближений до реальних систем моніторингу, але залишається простим для демонстрації.

Після відкриття вебінтерфейсу адміністратор може переглянути загальний стан системи, список контрольованих вузлів, їхні поточні статуси, час останньої перевірки та базові метрики. За потреби користувач може ініціювати перевірку вузлів або переглянути вже підготовлені результати, отримані під час попереднього циклу опитування. Це дозволяє швидко оцінити, які пристрої працюють у нормальному режимі, а які потребують уваги. Результати роботи системи можуть подаватися у вигляді таблиць, інформаційних карток, графіків або записів журналу подій. Таблиці зручні для перегляду переліку вузлів, картки дозволяють швидко побачити загальні показники графіку.

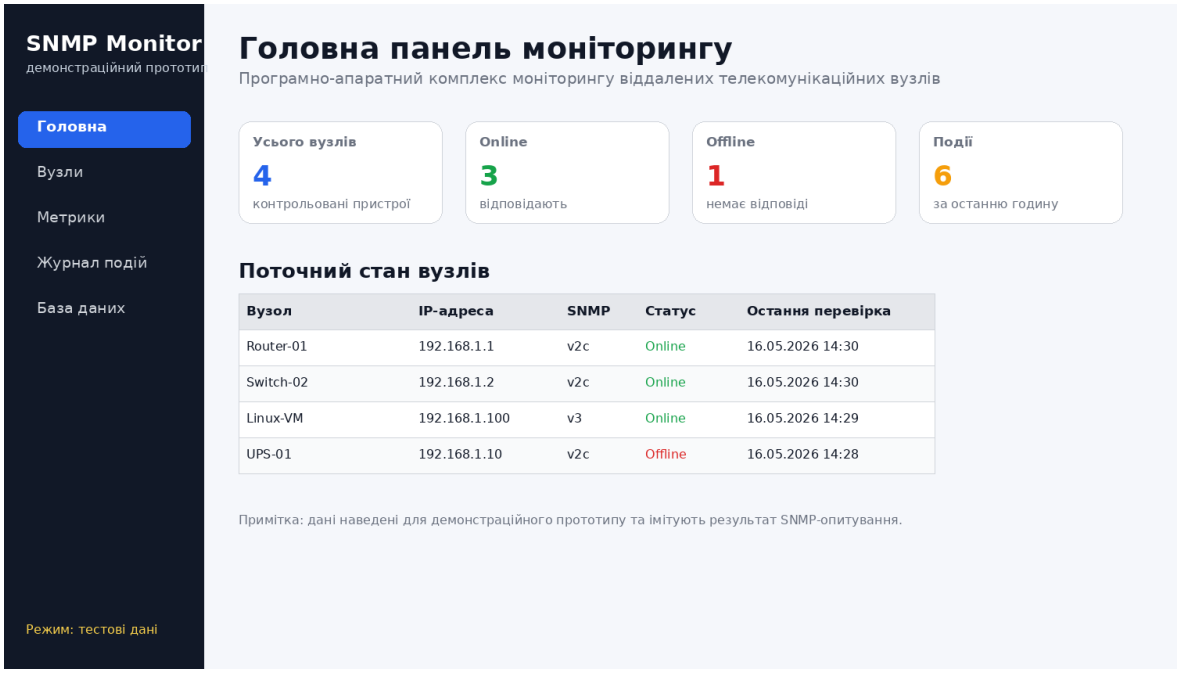


Рисунок 3.8 – Головна панель моніторингу віддалених телекомунікаційних вузлів

Головна панель, наведена на рисунку 3.8, призначена для швидкої оцінки стану системи. На ній адміністратор одразу бачить загальну кількість вузлів, кількість доступних і недоступних пристроїв, а також кількість подій. Такий екран зручний як стартова сторінка, бо він не перевантажує користувача деталями.

У нижній частині головної панелі показується коротка таблиця поточного стану вузлів. Для кожного пристрою наведено назву, IP-адресу, версію SNMP, статус і час останньої перевірки. Цієї інформації достатньо, щоб зрозуміти, який вузол потребує уваги.

Як завдання моніторингу на цьому екрані можна розглядати швидку перевірку поточного стану всієї інфраструктури. Адміністратор не переходить до кожного вузла окремо, а спочатку бачить узагальнену картину. Якщо всі вузли доступні, додаткових дій може не знадобитися. Якщо є вузол Offline, користувач переходить до детальнішого перегляду.

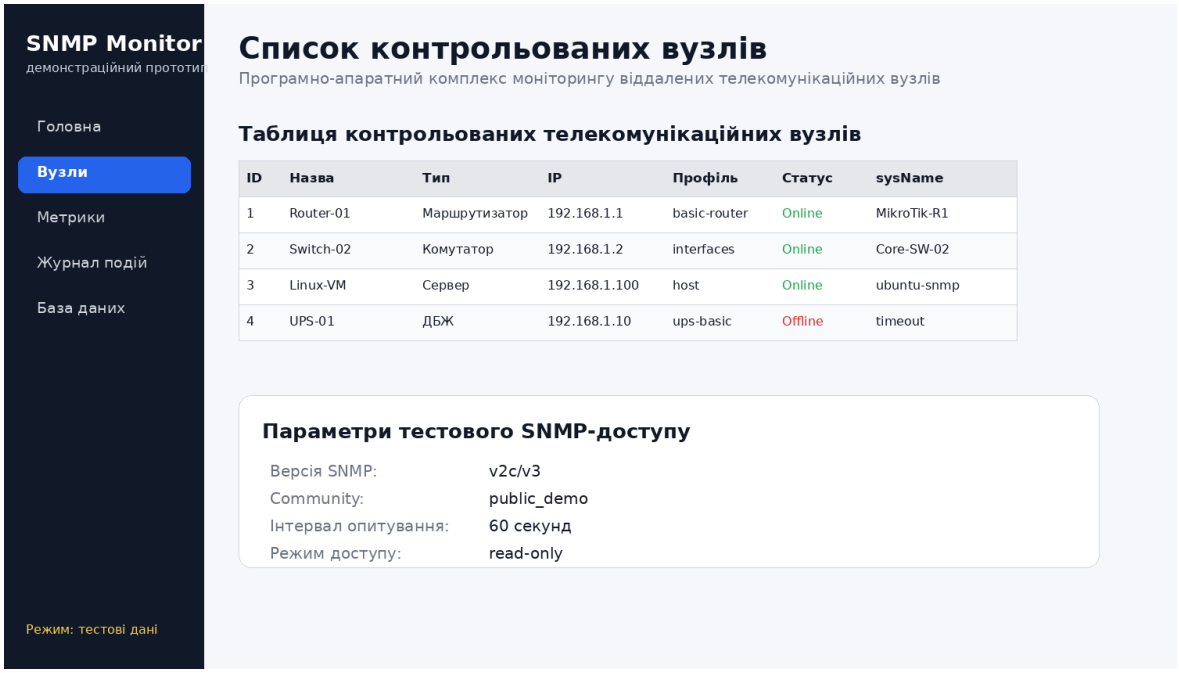


Рисунок 3.9 – Список контрольованих вузлів у вебінтерфейсі

Сторінка вузлів, зображена на рисунку 3.9, потрібна для перегляду детальнішої інформації про кожний пристрій. Тут можна побачити тип вузла, його IP-адресу, профіль SNMP-опитування, статус і службове ім’я. У повній системі така сторінка може також містити кнопку ручного запуску перевірки або форму додавання нового вузла.

Постановка завдання через цю сторінку може виглядати як вибір конкретного вузла для перевірки. Наприклад, якщо адміністратор бачить, що UPS-01 знаходиться у стані Offline, він може вибрати цей вузол і переглянути журнал подій або останній час перевірки. У реальній системі також можна додати повторне опитування.

Важливо, що інтерфейс не повинен бути складним. Для бакалаврської роботи достатньо показати основні елементи, які підтверджують логіку системи: перелік вузлів, статуси, базові параметри і можливість переходу до детальної інформації.

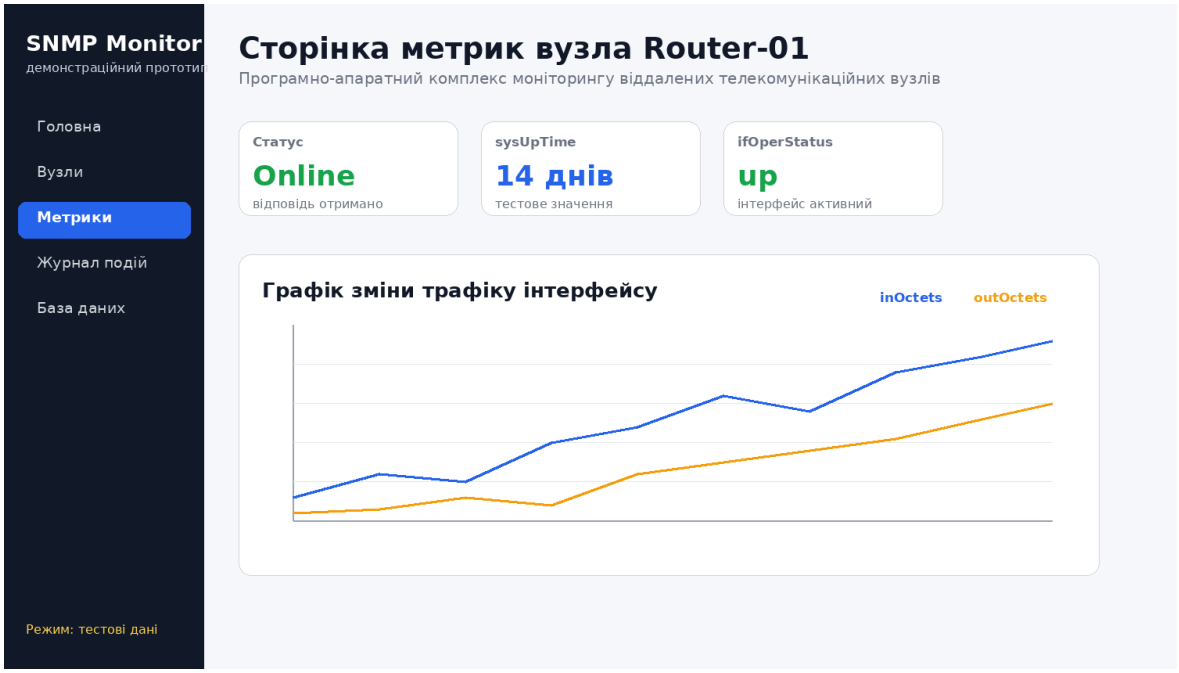


Рисунок 3.10 – Сторінка перегляду SNMP-метрик вузла

Сторінка метрик, наведена на рисунку 3.10, демонструє, як система може подавати інформацію про окремий вузол. У прикладі показано стан Router-01, значення sysUpTime, стан інтерфейсу та графік зміни трафіку. Навіть якщо дані у прототипі тестові, їх структура відповідає реальним SNMP-параметрам.

На цій сторінці адміністратор може виконувати більш конкретне завдання: оцінити, чи працює вузол стабільно. Якщо час роботи великий і статус Online, можна зробити висновок, що пристрій не перезавантажувався і доступний. Якщо трафік має різкі зміни або статус інтерфейсу змінюється, це може бути причиною для додаткової перевірки.

Графічне подання метрик важливе тому, що воно швидше сприймається, ніж набір чисел. Для реальної системи можна додати графіки трафіку, помилок інтерфейсів або часу відповіді. У демонстраційному прототипі достатньо одного графіка, який показує принцип візуалізації. Для реальної системи моніторингу можна додати графіки вхідного та вихідного трафіку, кількості помилок на інтерфейсах, часу відповіді вузла, завантаження каналів або зміни стану живлення.

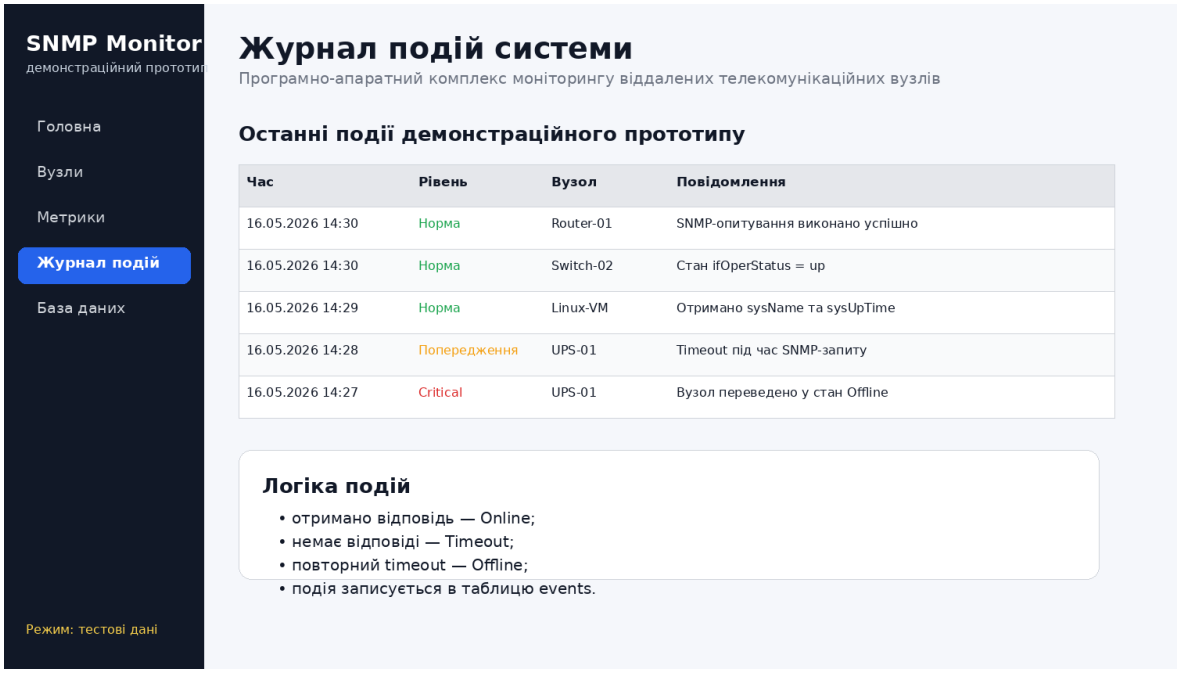


Рисунок 3.11 – Журнал подій системи моніторингу

Журнал подій на рисунку 3.11 показує історію роботи системи. У ньому можуть відображатися успішні перевірки, timeout, зміна статусу вузла, попередження та критичні події. Для адміністратора це важливий екран, бо саме журнал дозволяє зрозуміти, коли виникла проблема.

Постановка завдання у цьому випадку полягає не в запуску опитування, а в аналізі вже накопиченої інформації. Наприклад, якщо один вузол кілька разів переходив у стан Offline, адміністратор може зробити висновок про нестабільність каналу або обладнання. Якщо подія виникла лише один раз, це може бути короткочасний збій.

Таким чином, вебінтерфейс забезпечує не тільки перегляд поточного стану контрольованих вузлів, а й роботу з історією подій. Це є важливою перевагою порівняно з простою перевіркою доступності через ping, оскільки адміністратор отримує не лише відповідь про те, доступний вузол чи ні, а й додатковий контекст щодо його роботи. У системі можна побачити час останньої перевірки, поточний статус, отримані метрики та записи журналу подій.

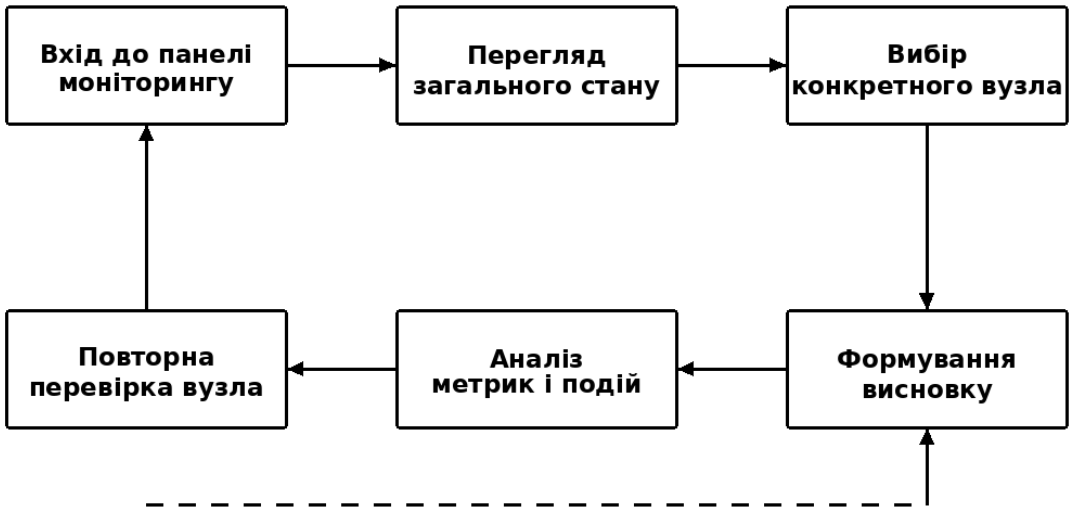


Рисунок 3.12 – Логіка роботи користувача у вебінтерфейсі

Рисунок 3.12 узагальнює послідовність дій користувача у вебінтерфейсі. Спочатку адміністратор відкриває головну панель, потім переглядає загальний стан, вибирає вузол, аналізує метрики і журнал подій, після чого формує висновок або запускає повторну перевірку.

У такій логіці вебінтерфейс виконує роль робочого місця адміністратора. Він не приховує технічну інформацію повністю, але подає її у зручній формі. Це особливо важливо для невеликих мереж, де адміністратор може одночасно відповідати за різні типи обладнання.

3.4 Практичні приклади застосування системи для моніторингу телекомунікаційних вузлів

Практичні приклади застосування потрібні для того, щоб показати, як запропонована система може використовуватися у типових умовах роботи невеликої мережевої або телекомунікаційної інфраструктури. Якщо у попередніх підрозділах розглядалися алгоритми, структура програмного забезпечення та веб-базований інтерфейс, то в цьому підрозділі доцільно показати, як ці елементи можуть працювати на конкретних прикладах. Такий підхід дозволяє краще

пояснити практичне призначення комплексу і показати, що система не обмежується лише загальною схемою або статичним макетом.

У межах демонстраційного прототипу розглянуто кілька умовних вузлів: маршрутизатор Router-01, комутатор Switch-02, сервер Linux-VM та джерело безперебійного живлення UPS-01. Такий набір вузлів обрано не випадково. Він дозволяє показати різні типи обладнання, які можуть бути присутні у віддаленому телекомунікаційному вузлі. Маршрутизатор відповідає за зв’язок із зовнішньою мережею, комутатор забезпечує підключення локальних пристроїв, сервер виконує прикладні або службові функції, а джерело безперебійного живлення підтримує роботу обладнання у разі проблем з електроживленням.

Перший приклад стосується маршрутизатора Router-01. Для такого пристрою важливо контролювати його доступність, час безперервної роботи, стан мережевих інтерфейсів і наявність помилок під час опитування. Якщо маршрутизатор стає недоступним, це може означати втрату зв’язку з усім віддаленим вузлом або окремим сегментом мережі. Тому для маршрутизатора важливо не лише бачити стан Online або Offline, а й мати можливість переглянути час останньої перевірки та основні службові параметри.

У демонстраційному інтерфейсі Router-01 має стан Online, а значення sysUpTime показує, що пристрій працює стабільно протягом тривалого часу. Це означає, що система отримала відповідь від вузла, коректно обробила SNMP-дані та відобразила результат у вебінтерфейсі адміністратора. У реальному режимі роботи до цього прикладу можна додати контроль завантаження інтерфейсів, кількості помилок передавання, стану портів і статистики трафіку.

Другий приклад стосується комутатора Switch-02. Для комутатора корисними є дані про стан інтерфейсів, активність портів, кількість переданих і прийнятих пакетів, а також наявність помилок або відкинутих кадрів. У невеликій мережі комутатор може працювати непомітно для користувача, але саме через нього проходить значна частина локального трафіку. Тому його стан

також потрібно контролювати, особливо якщо до нього підключені сервери, точки доступу або інші важливі пристрої.

Якщо один із портів комутатора переходить у стан down або кількість помилок різко зростає, система може сформувати подію Warning. У межах демонстраційного прототипу показано лише спрощений варіант, але сама логіка легко розширюється. Наприклад, можна додати контроль конкретних інтерфейсів, порогові значення для кількості помилок або попередження у разі перевищення певного рівня трафіку. Це дозволить використовувати систему не лише для перевірки доступності, а й для початкового аналізу якості роботи мережевих з'єднань.

Третій приклад пов'язаний із сервером Linux-VM. У демонстраційному стенді такий вузол використовується як умовний сервер, що може мати SNMP-агент або передавати тестові службові дані. Для сервера можна контролювати доступність, службове ім'я, час безперервної роботи та події, пов'язані з втратою зв'язку. Якщо сервер не відповідає на запит, система повинна зафіксувати timeout, змінити його стан на Offline і створити відповідний запис у журналі подій.

У реальному варіанті функціональність для сервера може бути розширена. До базових SNMP-параметрів можна додати контроль навантаження процесора, використання оперативної пам'яті, заповнення дискового простору, стану мережевого інтерфейсу та службових процесів. Проте в межах бакалаврської роботи достатньо показати загальний принцип: система отримує або імітує відповідь від вузла, обробляє її, оновлює стан і показує результат адміністратору.

Четвертий приклад стосується джерела безперебійного живлення UPS-01. Такий пристрій важливий для віддалених телекомунікаційних вузлів, тому що проблеми з живленням можуть призвести до відмови всього майданчика. Навіть якщо мережеве обладнання працює справно, відсутність резервного живлення або несправність UPS може стати причиною аварійного вимкнення пристроїв. Тому стан джерела живлення також доцільно включити до системи моніторингу.

У демонстраційному прототипі для UPS-01 може бути показана тестова подія, наприклад timeout або попередження про проблемний стан. Така подія демонструє, як система може повідомити адміністратора про потенційно небезпечну ситуацію. У реальному режимі для UPS можна контролювати рівень заряду батареї, режим роботи від мережі або батареї, температуру, стан навантаження та аварійні повідомлення, якщо такі параметри підтримуються пристроєм через SNMP.

Розглянуті приклади показують, що один і той самий програмно-апаратний комплекс може використовуватися для різних типів вузлів. Для маршрутизатора основний акцент робиться на доступності та мережевих параметрах, для комутатора на стані портів і трафіку, для сервера на службових параметрах операційної системи, а для UPS на контролі живлення. При цьому загальна логіка системи залишається однаковою: вузол додається до переліку, система отримує або імітує дані, обробляє результат, оновлює статус і формує подію за потреби.

Практичне значення таких прикладів полягає в тому, що вони показують можливість подальшого розширення прототипу. Якщо у майбутньому потрібно буде додати новий тип обладнання, наприклад точку доступу, мережеве сховище або контролер, це можна зробити без повної перебудови системи. Достатньо додати опис вузла, визначити набір параметрів для контролю та налаштувати спосіб отримання даних. Завдяки цьому система може поступово розвиватися від демонстраційного прототипу до більш повноцінного засобу моніторингу.

Таким чином, практичні приклади застосування підтверджують працездатність запропонованої логіки. Система може використовуватися для контролю різних типів віддалених телекомунікаційних вузлів, відображати їхній стан, фіксувати події та надавати адміністратору зрозумілу інформацію через веб-базований інтерфейс. Навіть у спрощеному демонстраційному вигляді такий комплекс дозволяє показати основні принципи SNMP-моніторингу та підготувати основу для подальшого підключення реального обладнання.

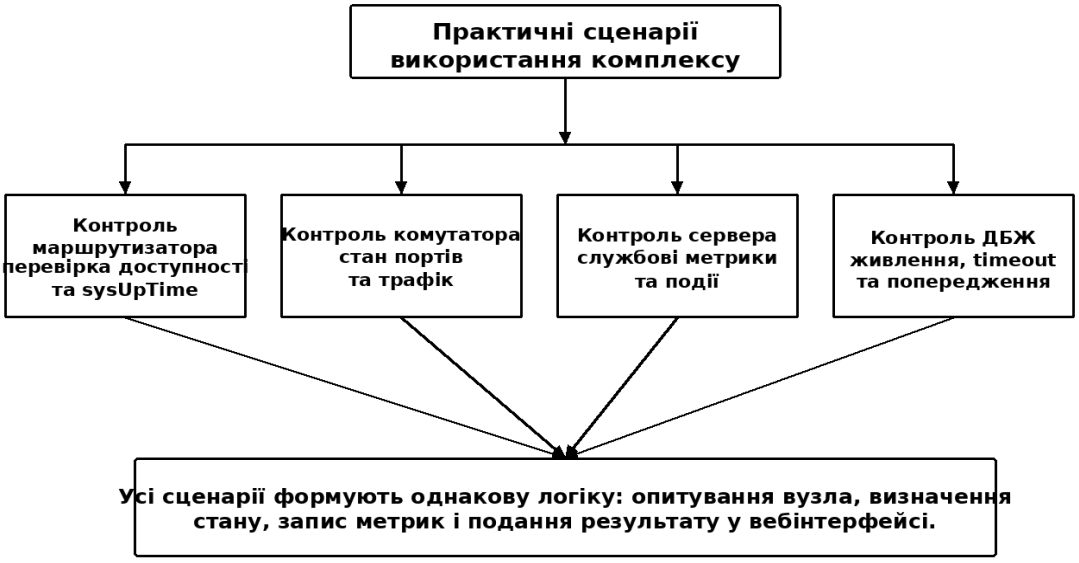


Рисунок 3.13 – Приклади застосування системи моніторингу

На рисунку 3.13 показано чотири практичні сценарії. Усі вони мають спільну логіку: система отримує інформацію про вузол, визначає його стан, записує результат і показує його у вебінтерфейсі. Відмінність полягає лише в тому, які саме параметри є найбільш важливими для конкретного типу обладнання.

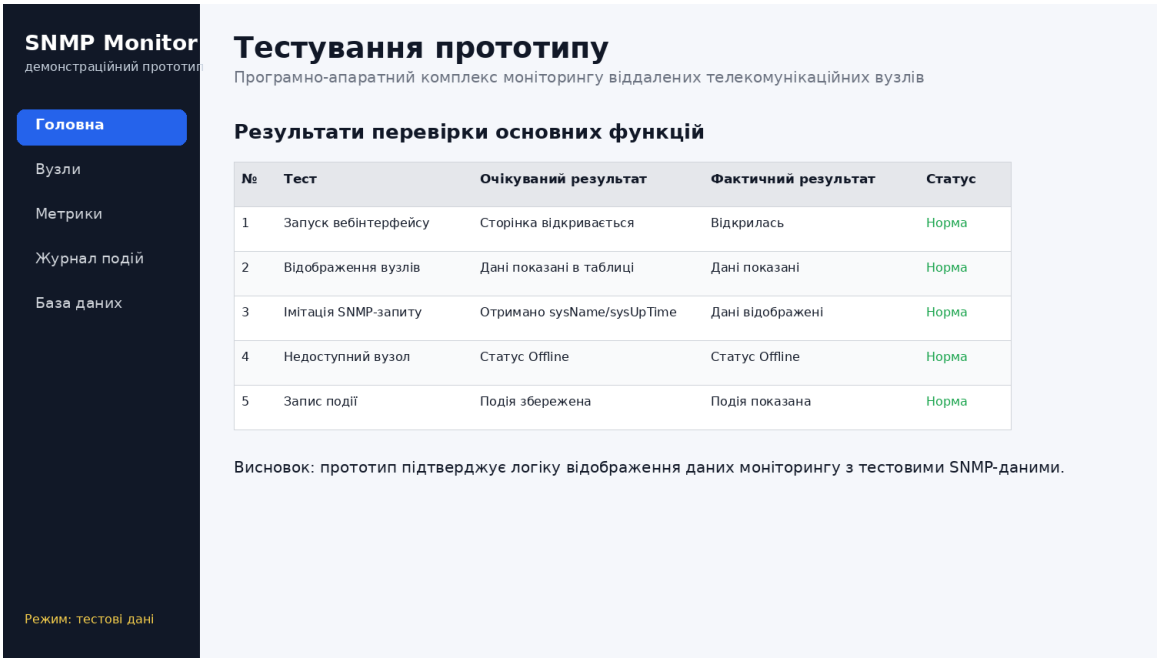


Рисунок 3.14 – Результати перевірки роботи демонстраційного прототипу

Результати перевірки, наведені на рисунку 3.14, показують основні функції демонстраційного прототипу. Перевіряється запуск вебінтерфейсу, відображення вузлів, імітація SNMP-запиту, оброблення недоступного вузла і запис події. Це не є повним промисловим тестуванням, але для дипломної роботи такий набір перевірок показує працездатність логіки.

Перший сценарій передбачає перевірку доступу до головної сторінки системи. Адміністратор відкриває вебінтерфейс і переконується, що панель моніторингу завантажується коректно. Цей етап є базовим, оскільки без доступу до головної сторінки неможливо використовувати інші функції комплексу, зокрема перегляд вузлів, метрик і журналу подій.

Другий сценарій пов’язаний із перевіркою відображення списку контрольованих вузлів. У цьому режимі система повинна показувати основні дані про кожен пристрій: назву, IP-адресу, поточний статус і час останньої перевірки. Такий список є основною робочою таблицею адміністратора, тому його коректне відображення має важливе значення для подальшого контролю мережевої інфраструктури.

У третьому сценарії перевіряється отримання або імітація SNMP-даних. Система підставляє тестові значення sysName і sysUpTime або отримує їх у результаті SNMP-запиту, після чого відображає ці параметри у відповідних блоках інтерфейсу. У реальному режимі на цьому етапі виконувався б запит до SNMP-агента пристрою, однак для демонстраційного прототипу достатньо використання підготовлених тестових значень.

Четвертий сценарій демонструє роботу системи у випадку недоступності вузла. Для пристрою UPS-01 моделюється ситуація timeout, після чого система змінює його стан на Offline і створює відповідний запис у журналі подій. Цей сценарій є важливим, оскільки показує поведінку комплексу не лише за нормальної роботи вузлів, а й у разі виникнення помилки або втрати зв’язку.

П’ятий сценарій стосується перевірки журналу подій. Система повинна не тільки сформувати подію всередині програми, а й відобразити її у вебінтерфейсі

адміністратора. Якщо повідомлення про зміну стану або помилку коректно, що комплекс може зберігати та показувати історію основних подій моніторингу.

Рисунок 3.15 – Приклад постановки завдання моніторингу через вебінтерфейс

Окремо можна розглянути приклад постановки завдання моніторингу. Адміністратор вибирає вузол, перевіряє його параметри і запускає повторне опитування або перегляд історії. У демонстраційному варіанті це завдання подається на рівні логіки інтерфейсу. У повній реалізації воно може бути оформлене у вигляді кнопки або форми.

Приклад із маршрутизатором показує нормальний сценарій. Вузол доступний, система отримує службові параметри, статус залишається Online, а в журналі створюється подія успішного опитування. Такий сценарій підтверджує, що базова функція моніторингу працює коректно.

Приклад із UPS-01 показує аварійний сценарій. Відповідь не отримано, система фіксує timeout і переводить вузол у стан Offline. У журналі подій з’являється запис, який може бути підставою для подальшої перевірки живлення, каналу зв’язку або налаштувань доступу.

Загалом наведені приклади показують, що навіть простий демонстраційний прототип може пояснити практичну цінність системи. Він не замінює повноцінний продукт, але демонструє, як дані збираються, зберігаються, обробляються і відображаються адміністратору.

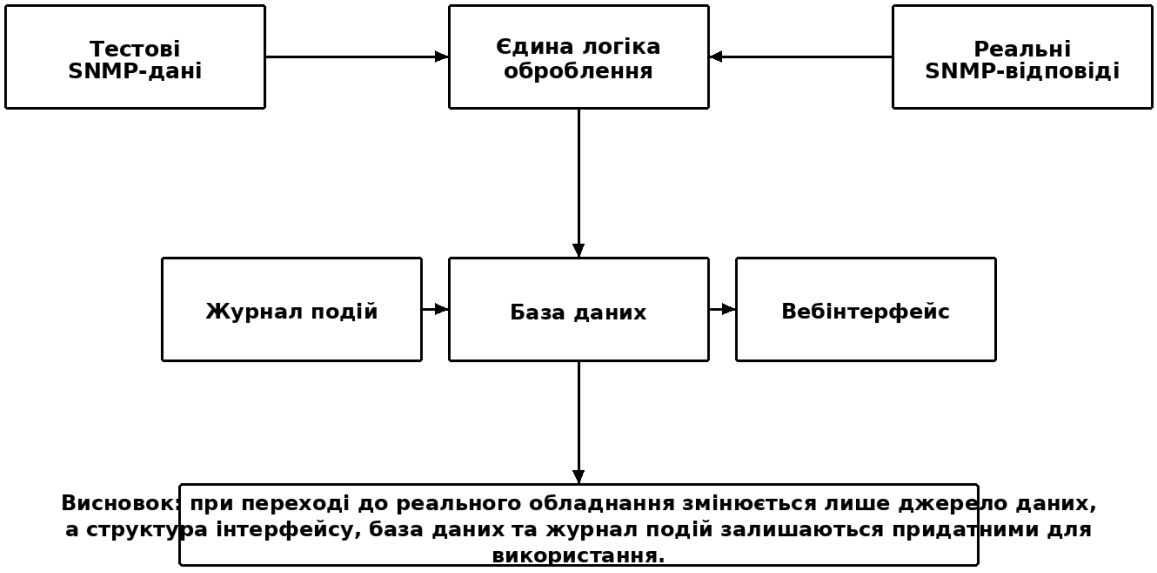


Рисунок 3.16 – Перехід від демонстраційного режиму до реального SNMP-опитування

Останній практичний момент стосується переходу від демонстраційного режиму до реального використання системи. У цій роботі для спрощення застосовано тестові SNMP-дані, оскільки головним завданням є не розгортання повноцінної промислової інфраструктури, а демонстрація логіки роботи програмно-апаратного комплексу. Такий підхід дозволяє перевірити структуру системи, роботу вебінтерфейсу, журналу подій, збереження даних і відображення станів вузлів без підключення до реального мережевого обладнання. Водночас структура комплексу побудована таким чином, щоб у майбутньому джерело тестових даних можна було замінити на реальні SNMP-відповіді.

Особливість запропонованого підходу полягає в тому, що демонстраційний режим не змінює загальну логіку роботи системи. У тестовому варіанті значення sysName, sysUpTime, статус вузла, події timeout або показники трафіку формуються заздалегідь або імітуються. У реальному режимі ці самі значення можуть надходити від SNMP-агентів, які працюють на маршрутизаторах, комутаторах, серверах або джерелах безперебійного живлення. Отже, змінюється лише спосіб отримання даних, а логіка їх оброблення, збереження та відображення залишається однаковою.

Рисунок 3.16 показує, що тестові дані і реальні SNMP-відповіді можуть надходити до одного й того самого модуля оброблення. Для бази даних і вебінтерфейсу не має принципового значення, чи значення було отримано від фізичного маршрутизатора, чи сформовано як демонстраційний приклад. Важливо лише те, щоб формат даних був однаковим. Наприклад, якщо модуль оброблення отримує назву вузла, IP-адресу, статус, час перевірки та значення метрик, то джерело цих значень може бути різним. У демонстраційному режимі це підготовлений набір даних, а в реальному режимі відповідь SNMP-агента.

Такий підхід значно полегшує подальше доопрацювання системи. Спочатку можна перевірити вебінтерфейс, журнал подій, структуру бази даних і сценарії роботи на тестових вузлах. Це дозволяє переконатися, що таблиця вузлів коректно відображає стан обладнання, сторінка метрик показує потрібні значення, а журнал подій фіксує зміну станів Online, Offline, Timeout або Warning. Після цього достатньо додати реальний модуль SNMP-опитування, який буде передавати значення у вже підготовлену структуру системи.

У разі переходу до реального режиму роботи потрібно буде виконати кілька додаткових дій. Насамперед необхідно налаштувати SNMP-агентів на мережевому обладнанні. Для кожного пристрою потрібно визначити версію SNMP, параметри доступу, дозволені IP-адреси серверів моніторингу та перелік OID, які будуть використовуватися для отримання даних. Для навчального або закритого середовища може бути достатньо SNMPv2c у режимі read-only, однак

для реальної експлуатації доцільніше використовувати SNMPv3, оскільки він підтримує автентифікацію та шифрування.

Також необхідно визначити список контрольованих параметрів. Для маршрутизатора або комутатора це можуть бути назва пристрою, час безперервної роботи, стан інтерфейсів, вхідний і вихідний трафік, кількість помилок або відкинутих пакетів. Для сервера можна контролювати службові параметри системи, доступність SNMP-агента та додаткові показники, які підтримує його конфігурація. Для джерела безперебійного живлення важливими можуть бути стан батареї, режим живлення, рівень заряду та повідомлення про аварійні події. Усі ці дані можуть бути додані до системи без зміни загального принципу її роботи.

Окремо потрібно підібрати інтервал опитування. Якщо інтервал буде занадто малим, система може створювати зайве навантаження на мережу та пристрої. Якщо інтервал буде занадто великим, адміністратор може отримувати інформацію із запізненням. Для демонстраційного прототипу цей параметр не має критичного значення, оскільки дані є тестовими. Проте у реальній системі інтервал опитування повинен залежати від важливості вузла, швидкості зміни параметрів і допустимого часу реакції на несправність.

Для практичного впровадження також потрібно врахувати обробку помилок. У тестовому режимі помилка timeout може бути просто підготовленим прикладом. У реальному середовищі timeout може виникати через втрату зв’язку, неправильні параметри доступу, вимкнений SNMP-agent, блокування порту firewall або перевантаження пристрою. Тому система повинна не лише показувати стан Offline, а й зберігати подію з поясненням причини. Це дає змогу адміністратору швидше визначити, що саме потрібно перевірити: доступність мережі, налаштування SNMP або стан самого обладнання.

Важливим напрямом розвитку є автоматичний планувальник опитування. У демонстраційному варіанті адміністратор може запускати перевірку вручну або переглядати заздалегідь підготовлені результати. У реальному режимі такий

підхід є незручним, оскільки адміністратор не повинен постійно вручну запускати перевірки. Планувальник може періодично формувати завдання опитування, передавати їх модулю SNMP-collector, а потім оновлювати дані в базі та вебінтерфейсі. У такому випадку адміністратор лише переглядає результати та реагує на події.

Ще одним можливим покращенням є використання різних інтервалів опитування для різних типів вузлів. Наприклад, маршрутизатор або основний комутатор можна перевіряти частіше, оскільки від них залежить доступність усього сегмента мережі. Менш критичні вузли можна перевіряти рідше. Якщо пристрій переходить у стан Warning або Offline, система може тимчасово збільшити частоту його перевірки. Такий підхід дозволить краще реагувати на проблеми і водночас не створювати зайве навантаження.

Під час переходу до реального використання важливо також зберегти принцип безпечної роботи. Система моніторингу не повинна змінювати налаштування мережевого обладнання. Для цієї роботи достатньо режиму читання, тобто отримання параметрів без використання SNMP SET. Це зменшує ризик випадкової зміни конфігурації пристроїв і робить систему безпечнішою для використання в навчальному або тестовому середовищі. У разі реального впровадження доступ до SNMP потрібно обмежувати за IP-адресами, а параметри доступу не слід зберігати у відкритому вигляді.

З точки зору бази даних перехід до реального режиму також не потребує повної перебудови. Таблиці, які зберігають вузли, метрики та події, можуть використовуватися і для тестових, і для реальних даних. У таблиці вузлів зберігається опис пристрою, у таблиці метрик отримані значення, а в таблиці подій повідомлення про зміну станів або помилки. Якщо система буде розширюватися, до цієї структури можна додати таблиці користувачів, ролей доступу, шаблонів OID, налаштувань опитування або правил сповіщення.

Для вебінтерфейсу перехід до реальних SNMP-відповідей також не повинен вимагати значних змін. Інтерфейс продовжує працювати з даними, які

вже збережені та оброблені. Це означає, що сторінка списку вузлів, картки стану, сторінка метрик і журнал подій можуть залишатися такими самими. Змінюється лише джерело оновлення даних. Завдяки цьому демонстраційний інтерфейс не є просто статичним макетом, а може розглядатися як основа для подальшого розвитку системи.

У дипломній роботі такий спосіб подання є зручним, оскільки він не створює враження завершеного промислового продукту, але показує реалістичний шлях розвитку прототипу. Система спочатку демонструє логіку роботи на тестових даних, а потім може бути доповнена реальним SNMP-collector. Це відповідає формату бакалаврської роботи, де основна мета полягає у проєктуванні, описі архітектури та демонстрації працездатної ідеї, а не у створенні повноцінної промислової платформи моніторингу.

Практична цінність такого підходу полягає у поступовості розроблення та перевірки системи. Спочатку доцільно перевірити структуру даних і вебінтерфейс, тобто переконатися, що система правильно відображає список вузлів, їхні статуси, базові метрики та записи журналу подій. На цьому етапі можна використовувати тестові SNMP-дані, які імітують відповіді від реальних пристроїв. Це дозволяє швидко перевірити зовнішню логіку роботи комплексу без складного налаштування мережевого обладнання.

Після перевірки структури даних і вебінтерфейсу можна додати логіку оброблення подій. На цьому етапі система повинна не лише показувати отримані значення, а й визначати, що означає конкретний результат опитування. Наприклад, якщо вузол відповідає на запит, він отримує стан Online, якщо відповідь відсутня — формується подія Timeout або Offline. Якщо з’являється попереджувальна ситуація, система може створити подію Warning. Такий підхід дозволяє поступово перейти від простого відображення даних до більш змістовного аналізу стану вузлів. Лише після цього доцільно підключати реальне SNMP-опитування.

3.5 Висновок до розділу 3

У третьому розділі було розглянуто алгоритмічне та програмне забезпечення демонстраційного прототипу системи моніторингу віддалених телекомунікаційних вузлів. Основну увагу приділено не програмному коду, а логіці роботи системи, послідовності виконання основних процедур, структурі програмних модулів, веб-базованому інтерфейсу та практичним прикладам застосування комплексу.

Було описано основні алгоритмічні процедури, які забезпечують роботу системи моніторингу. До них належать завантаження переліку вузлів, вибір пристрою для перевірки, формування SNMP-запиту або використання тестових SNMP-даних, очікування відповіді, оброблення результатів, визначення стану вузла, запис метрик до бази даних, формування подій і оновлення вебінтерфейсу адміністратора. Така послідовність дозволяє показати повний шлях проходження даних від контрольованого вузла до користувача системи.

Окремо було розглянуто структуру програмного забезпечення демонстраційного прототипу. У складі системи виділено модуль SNMP-опитування, модуль тестових даних, модуль оброблення результатів, базу даних, журнал подій, сервер застосунку та вебінтерфейс. Такий поділ є зручним, оскільки кожна частина виконує окрему функцію і може бути змінена або розширена без повної перебудови всієї системи. Наприклад, у майбутньому модуль тестових даних можна замінити реальним SNMP-опитуванням, залишивши без змін базу даних, журнал подій і вебінтерфейс.

Було показано, що веб-базований інтерфейс є важливою частиною комплексу, оскільки саме через нього адміністратор отримує доступ до результатів моніторингу. Інтерфейс дозволяє переглядати загальний стан системи, список контрольованих вузлів, базові SNMP-метрики, графічне подання окремих показників і журнал подій. Завдяки цьому адміністратор може оцінювати стан вузлів не тільки за поточним статусом Online або Offline, а й за

додатковими даними, зокрема часом останньої перевірки, отриманими метриками та історією подій.

У практичній частині розділу наведено приклади застосування системи для різних типів телекомунікаційних вузлів: маршрутизатора, комутатора, сервера та джерела безперебійного живлення. Такі приклади показують, що запропонований підхід може використовуватися для контролю різнорідного обладнання. Для маршрутизатора важливими є доступність і час роботи, для комутатора — стан інтерфейсів і трафік, для сервера — доступність і службові параметри, а для ДБЖ — стан живлення та попереджувальні події.

Результати опису демонстраційного прототипу показують, що система здатна виконувати базові функції моніторингу: працювати з переліком вузлів, отримувати або імітувати SNMP-дані, визначати стан пристроїв, фіксувати події, зберігати результати та подавати їх у вебінтерфейсі. Хоча прототип не є промисловою системою моніторингу, він достатньо повно демонструє основну логіку роботи програмно-апаратного комплексу.

Таким чином, у третьому розділі було підтверджено практичну можливість реалізації запропонованого підходу до побудови програмно-апаратного комплексу моніторингу віддалених телекомунікаційних вузлів. Описані алгоритми, структура програмного забезпечення, вебінтерфейс і приклади застосування створюють основу для подальшого розвитку системи та її поступового переходу від демонстраційного прототипу до реального використання.

У межах розділу показано, що навіть спрощена реалізація дає змогу відобразити основні процеси моніторингу: отримання або імітацію SNMP-даних, визначення стану вузла, запис метрик, формування подій і подання результатів адміністратору. Це підтверджує, що обрана структура є придатною для демонстрації роботи системи та може бути використана як базова модель для подальшого доопрацювання.

ВИСНОВКИ

У першому розділі було розглянуто сучасні технології, методи та програмні засоби моніторингу віддалених телекомунікаційних вузлів. Визначено, що такі вузли можуть містити маршрутизатори, комутатори, сервери, джерела безперебійного живлення та інші пристрої, які забезпечують стабільну роботу мережевої інфраструктури.

Проаналізовано основні параметри, які доцільно контролювати під час моніторингу телекомунікаційного вузла. До них належать доступність пристрою, час безперервної роботи, стан мережевих інтерфейсів, обсяг вхідного та вихідного трафіку, кількість помилок передавання даних, стан живлення та події, що виникають у процесі роботи обладнання.

Розглянуто основні методи моніторингу мережевого обладнання, зокрема ICMP, SNMP polling, SNMP traps, Syslog, агентний моніторинг та API виробників. Встановлено, що для задачі контролю стану віддалених телекомунікаційних вузлів доцільно використовувати протокол SNMP, оскільки він підтримується великою кількістю мережевих пристроїв і дозволяє отримувати службові параметри обладнання.

Також виконано огляд програмних засобів моніторингу, серед яких Zabbix, Grafana, Prometheus, Telegraf, LibreNMS, Checkmk та PRTG. Визначено, що готові системи мають широкі можливості, але для бакалаврської роботи доцільно розглядати спрощений програмно-апаратний комплекс, який демонструє основну логіку SNMP-моніторингу.

У другому розділі було виконано проєктування програмно-апаратного комплексу моніторингу віддалених телекомунікаційних вузлів з використанням SNMP. Розглянуто загальну структуру комплексу, його основні компоненти та взаємозв’язки між ними.

Запропонована архітектура передбачає наявність контрольованих телекомунікаційних вузлів, модуля SNMP-опитування, бази даних, модуля аналізу, журналу подій та веб-базованого інтерфейсу адміністратора. Такий

підхід дозволяє розділити функції системи між окремими компонентами та зробити її більш зрозумілою для подальшої реалізації.

У розділі визначено основні функції, які може виконувати система у своїх компонентах. До них належать ведення переліку вузлів, отримання базових SNMP-параметрів, зберігання історії метрик, визначення стану вузла, формування подій і подання інформації адміністратору через вебінтерфейс.

Окремо розглянуто принципи автономної роботи комплексу. Показано, що система може самостійно виконувати перевірку вузлів, обробляти отримані або тестові дані, змінювати статус пристрою залежно від результату опитування та формувати записи у журналі подій без постійного втручання адміністратора.

У третьому розділі було розглянуто алгоритмічне та програмне забезпечення демонстраційного прототипу програмно-апаратного комплексу моніторингу віддалених телекомунікаційних вузлів. Основну увагу приділено практичній логіці роботи системи, але без наведення повного програмного коду, який може бути винесений у додатки.

Було описано алгоритми роботи комплексу у вигляді псевдокоду. Розглянуто послідовність завантаження переліку вузлів, виконання SNMP-опитування або отримання тестових даних, перевірки відповіді від вузла, оновлення статусу Online/Offline, запису метрик і формування подій у журналі.

Описано структуру та склад програмного забезпечення демонстраційного прототипу. До його основних частин належать модуль роботи з вузлами, модуль SNMP-опитування, модуль зберігання даних, модуль обробки подій і веб-базований інтерфейс адміністратора. Такий поділ дозволяє логічно розмежувати функції системи та спростити її подальше розширення.

Розглянуто веб-базований інтерфейс, який забезпечує доступ адміністратора до системи моніторингу віддалених телекомунікаційних вузлів. Інтерфейс дозволяє переглядати загальний стан системи, список контрольованих пристроїв, їхні поточні статуси, базові SNMP-метрики, графічне подання окремих показників і журнал подій. Завдяки цьому адміністратор може

отримувати інформацію про стан вузлів у зручному вигляді без необхідності безпосередньо підключатися до кожного маршрутизатора, комутатора, сервера або іншого пристрою окремо.

Окрему увагу приділено тому, що вебінтерфейс працює з уже підготовленими даними, які були отримані або зімітовані системою, оброблені та збережені у базі даних. Такий підхід дозволяє відокремити процес SNMP-опитування від процесу відображення результатів. У межах демонстраційного прототипу для перевірки роботи використано тестові дані, які імітують результати SNMP-опитування. Це дало змогу показати логіку роботи системи без підключення до реального мережевого обладнання та без складного налаштування SNMP-агентів.

Також наведено практичні приклади застосування системи для контролю маршрутизатора, комутатора, сервера та джерела безперебійного живлення. Такі приклади показують, що один і той самий програмно-апаратний комплекс може використовуватися для різних типів телекомунікаційних вузлів. Для маршрутизатора важливими є доступність і час роботи, для комутатора — стан інтерфейсів і трафік, для сервера — доступність та службові параметри, а для джерела безперебійного живлення — стан живлення і попереджувальні події.
Результати демонстрації показують, що запропонований прототип дозволяє відобразити основну логіку роботи системи моніторингу: визначити стан вузлів, зафіксувати події, зберегти отримані або тестові метрики та надати адміністратору інформацію у зрозумілому вигляді.

Навіть у спрощеному вигляді система демонструє повний шлях проходження даних: від контрольованого вузла або тестового джерела до модуля оброблення, бази даних, журналу подій і веб-базованого інтерфейсу. Таким чином, третій розділ показує практичну сторону запропонованого програмно-апаратного комплексу. Описаний демонстраційний прототип не є повноцінною промисловою системою моніторингу, однак він дозволяє перевірити основну ідею роботи.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. IETF. RFC 3411. An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks. 2002. URL: https://datatracker.ietf.org/doc/html/rfc3411 (дата звернення: 5.05.2026).

2. IETF. RFC 3412. Message Processing and Dispatching for the Simple Network Management Protocol (SNMP). 2002. URL: https://datatracker.ietf.org/doc/html/rfc3412 (дата звернення: 5.05.2026).

3. IETF. RFC 3413. Simple Network Management Protocol (SNMP) Applications. 2002. URL: https://datatracker.ietf.org/doc/html/rfc3413 (дата звернення: 20.05.2026).

4. IETF. RFC 3414. User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3). 2002. URL: https://datatracker.ietf.org/doc/html/rfc3414 (дата звернення: 5.05.2026).

5. IETF. RFC 3415. View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP). 2002. URL: https://datatracker.ietf.org/doc/html/rfc3415 (дата звернення: 5.05.2026).

6. IETF. RFC 3416. Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP). 2002. URL: https://datatracker.ietf.org/doc/html/rfc3416 (дата звернення: 5.05.2026).

7. IETF. RFC 3417. Transport Mappings for the Simple Network Management Protocol (SNMP). 2002. URL: https://datatracker.ietf.org/doc/html/rfc3417 (дата звернення: 5.05.2026).

8. IETF. RFC 3418. Management Information Base (MIB) for the Simple Network Management Protocol (SNMP). 2002. URL: https://datatracker.ietf.org/doc/html/rfc3418 (дата звернення:5.05.2026).

9. IETF. RFC 2863. The Interfaces Group MIB. 2000. URL: https://datatracker.ietf.org/doc/html/rfc2863 (дата звернення: 5.05.2026).

10. IETF. RFC 6353. Transport Layer Security (TLS) Transport Model for the Simple Network Management Protocol (SNMP). 2011. URL: https://datatracker.ietf.org/doc/html/rfc6353 (дата звернення: 5.05.2026).

11. Net-SNMP Project. Net-SNMP: Simple Network Management Protocol tools and libraries. Official project website. URL: https://www.net-snmp.org/ (дата звернення: 20.05.2026).

12. Net-SNMP Project. snmpwalk manual page: retrieve a subtree of management values using SNMP GETNEXT requests. URL: https://www.net-snmp.org/docs/man/snmpwalk.html (дата звернення: 5.05.2026).

13. Net-SNMP Project. Net-SNMP Tutorial: SNMPv3 Options. URL: https://www.net-snmp.org/tutorial/tutorial-5/commands/snmpv3.html (дата звернення: 5.05.2026).

14. LeXtudio Inc. PySNMP 7.1 Documentation. URL: https://docs.lextudio.com/pysnmp/v7.1/ (дата звернення: 5.05.2026).

15. Python Software Foundation. sqlite3 - DB-API 2.0 interface for SQLite databases. Python 3 Documentation. URL: https://docs.python.org/3/library/sqlite3.html (дата звернення: 5.05.2026).

16. SQLite Consortium. SQLite Documentation. URL: https://www.sqlite.org/docs.html (дата звернення: 5.05.2026).

17. Python Software Foundation. asyncio - Asynchronous I/O. Python 3 Documentation. URL: https://docs.python.org/3/library/asyncio.html (дата звернення: 5.05.2026).

18. Zabbix LLC. SNMP agent item type. Zabbix Documentation. URL: https://www.zabbix.com/documentation/current/en/manual/config/items/itemtypes/snmp (дата звернення: 5.05.2026).

19. Zabbix LLC. SNMP trap item type. Zabbix Documentation. URL: https://www.zabbix.com/documentation/current/en/manual/config/items/itemtypes/snmptrap (дата звернення: 5.05.2026).

20. Grafana Labs. Prometheus data source. Grafana Documentation. URL: https://grafana.com/docs/grafana/latest/datasources/prometheus/ (дата звернення: 5.05.2026).

21. Grafana Labs. Grafana Alerting. Grafana Documentation. URL: https://grafana.com/docs/grafana/latest/alerting/ (дата звернення: 5.05.2026).

22. Prometheus Authors. SNMP Exporter for Prometheus. GitHub repository. URL: https://github.com/prometheus/snmp_exporter (дата звернення: 5.05.2026).

23. InfluxData. Telegraf SNMP Input Plugin. GitHub repository. URL: https://github.com/influxdata/telegraf/tree/master/plugins/inputs/snmp (дата звернення: 5.05.2026).

24. InfluxData. Telegraf SNMP Trap Input Plugin. GitHub repository. URL: https://github.com/influxdata/telegraf/tree/master/plugins/inputs/snmp_trap (дата звернення: 5.05.2026).

25. LibreNMS. Auto-discovery Setup. LibreNMS Documentation. URL: https://docs.librenms.org/Extensions/Auto-Discovery/ (дата звернення: 5.05.2026).

26. LibreNMS. SNMP Configuration Examples. LibreNMS Documentation. URL: https://docs.librenms.org/Support/SNMP-Configuration-Examples/ (дата звернення: 5.05.2026).

27. Checkmk GmbH. Monitoring via SNMP: Monitoring of SNMP devices with Checkmk. URL: https://docs.checkmk.com/latest/en/snmp.html (дата звернення: 5.05.2026).

28. OpenNMS. SNMP Profiles. OpenNMS Meridian Documentation. URL: https://docs.opennms.com/meridian/2024/operation/deep-dive/provisioning/snmp-profile.html (дата звернення: 5.05.2026).

29. Paessler GmbH. Monitoring via SNMP. PRTG Manual. URL: https://www.paessler.com/manuals/prtg/monitoring_via_snmp (дата звернення: 5.05.2026).

30. Paessler GmbH. SNMP Trap Receiver Sensor. PRTG Manual. URL: https://www.paessler.com/manuals/prtg/snmp_trap_receiver_sensor (дата звернення: 5.05.2026).

31. MikroTik. SNMP. RouterOS Documentation. URL: https://help.mikrotik.com/docs/spaces/ROS/pages/8978519/SNMP (дата звернення: 5.05.2026).

32. Cisco Systems. SNMP Configuration Guide, Cisco IOS XE 17. URL: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/snmp/configuration/xe-17-x/snmp-xe-17-book.html (дата звернення: 5.05.2026).

33. Juniper Networks. Understand SNMP Implementation in Junos OS. URL: https://www.juniper.net/documentation/us/en/software/junos/network-mgmt/topics/topic-map/understand-snmp-implementation-in-junos-os.html (дата звернення: 5.05.2026).

34. Fortinet. SNMP. FortiGate / FortiOS 8.0.0 Administration Guide. URL: https://docs.fortinet.com/document/fortigate/8.0.0/administration-guide/62595/snmp (дата звернення: 5.05.2026).

35. Hewlett Packard Enterprise / Aruba. Configuring SNMP on AOS-CX. URL: https://help.centralon-prem.arubanetworks.com/2.5.4/documentation/online_help/content/nms-on-prem/aos-cx/cfg/conf-cx-snmp.htm (дата звернення: 5.05.2026).

36. NIST. Special Publication 800-137. Information Security Continuous Monitoring (ISCM) for Federal Information Systems and Organizations. 2011. URL: https://csrc.nist.gov/pubs/sp/800/137/final (дата звернення: 5.05.2026).

37. NIST. Special Publication 800-53 Revision 5. Security and Privacy Controls for Information Systems and Organizations. 2020, updated. URL: https://csrc.nist.gov/pubs/sp/800/53/r5/upd1/final (дата звернення: 5.05.2026).

38. NIST. The NIST Cybersecurity Framework (CSF) 2.0. 2024. URL: https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.29.pdf (дата звернення: 5.05.2026).

39. CISA. Cross-Sector Cybersecurity Performance Goals, Version 2.0. 2025. URL:https://www.cisa.gov/sites/default/files/2025-12/CPG_Report_2.0_508c.pdf (дата звернення: 5.05.2026).

40. ENISA. ENISA Threat Landscape 2025. European Union Agency for Cybersecurity. 2025. URL: https://www.enisa.europa.eu/publications/enisa-threat-landscape-2025 (дата звернення: 5.05.2026).

41. IETF. RFC 5424. The Syslog Protocol. 2009. URL: https://datatracker.ietf.org/doc/html/rfc5424 (дата звернення: 5.05.2026).

42. Pallets Projects. Flask Documentation 3.1.x. URL: https://flask.palletsprojects.com/ (дата звернення: 5.05.2026).

43. Bootstrap Team. Bootstrap 5.3 Documentation. URL: https://getbootstrap.com/docs/5.3/getting-started/introduction/(дата звернення: 5.05.2026).

44. Chart.js. Chart.js Documentation. URL: https://www.chartjs.org/docs/ (дата звернення: 5.05.2026).

45. MDN Web Docs. HTML: HyperText Markup Language. URL: https://developer.mozilla.org/en-US/docs/Web/HTML (дата звернення: 5.05.2026).

46. SQLAlchemy. SQLAlchemy 2.0 Documentation. URL: https://docs.sqlalchemy.org/ (дата звернення: 5.05.2026).

47. PostgreSQL Global Development Group. PostgreSQL Documentation. URL: https://www.postgresql.org/docs/ (дата звернення: 5.05.2026).

48. OWASP Foundation. OWASP Top 10:2025. URL: https://owasp.org/Top10/2025/en/ (дата звернення: 5.05.2026).

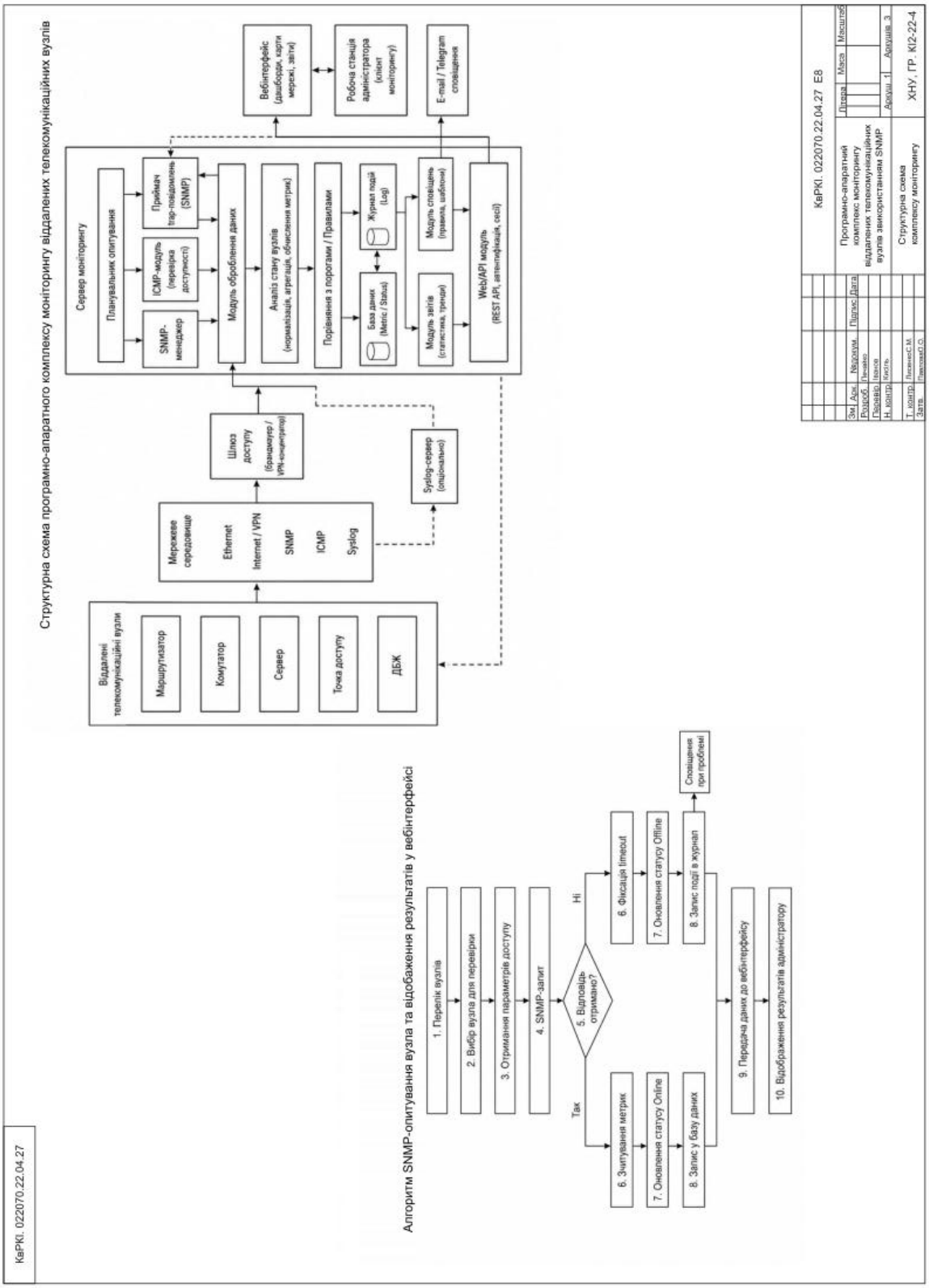
49. OWASP Foundation. OWASP Top Ten Web Application Security Risks. URL: https://owasp.org/www-project-top-ten/ (дата звернення: 5.05.2026).

50. Mozilla Developer Network. Structuring content with HTML. MDN Web Docs. URL:https://developer.mozilla.org/enUS/docs/Learn_web_development/Core/Structuring_content (дата звернення: 5.05.2026).

ДОДАТОК А

(обов’язковий)

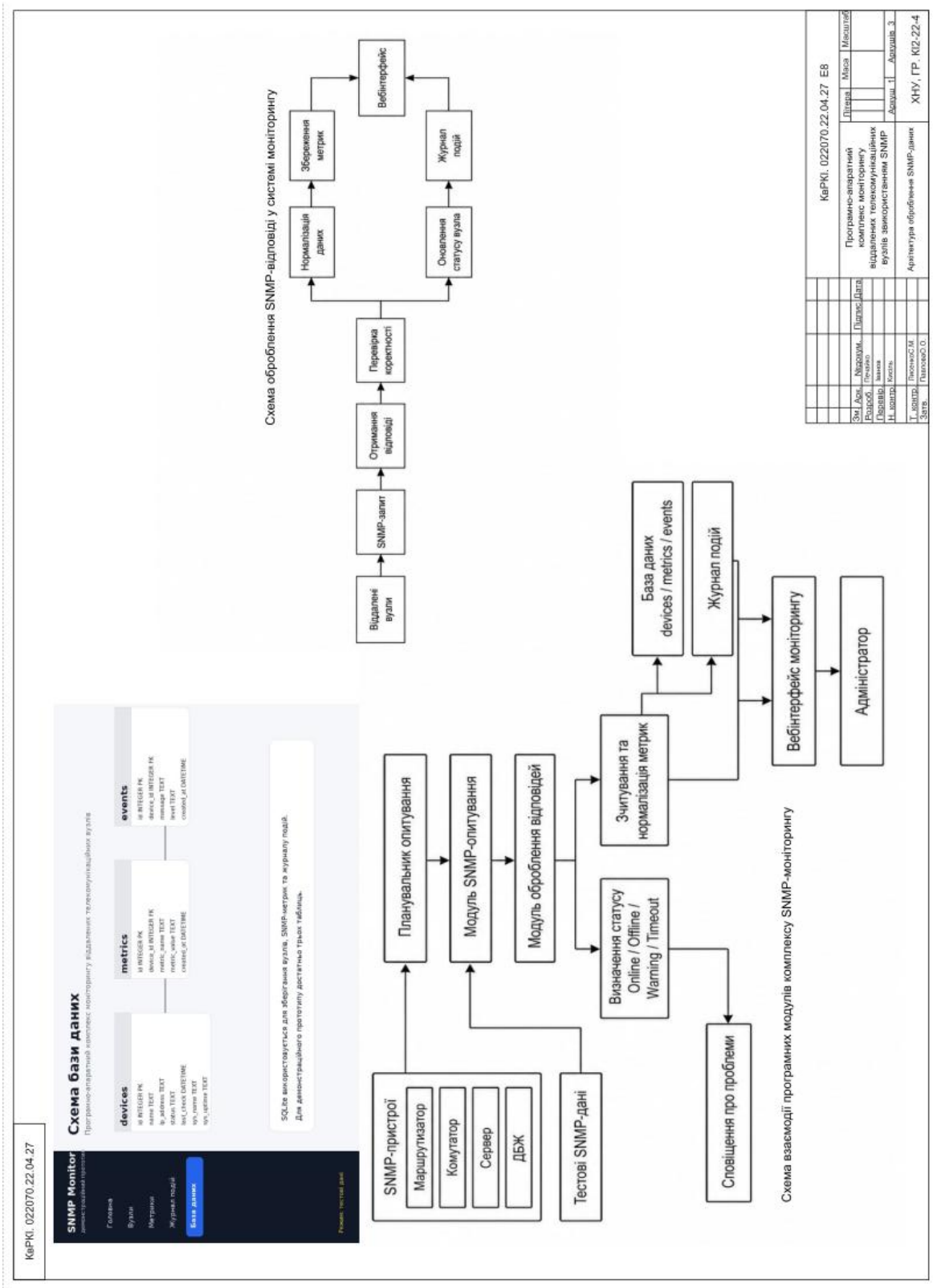
Копія креслення «Структурна схема комплексу моніторингу»



ДОДАТОК Б

(обов’язковий)

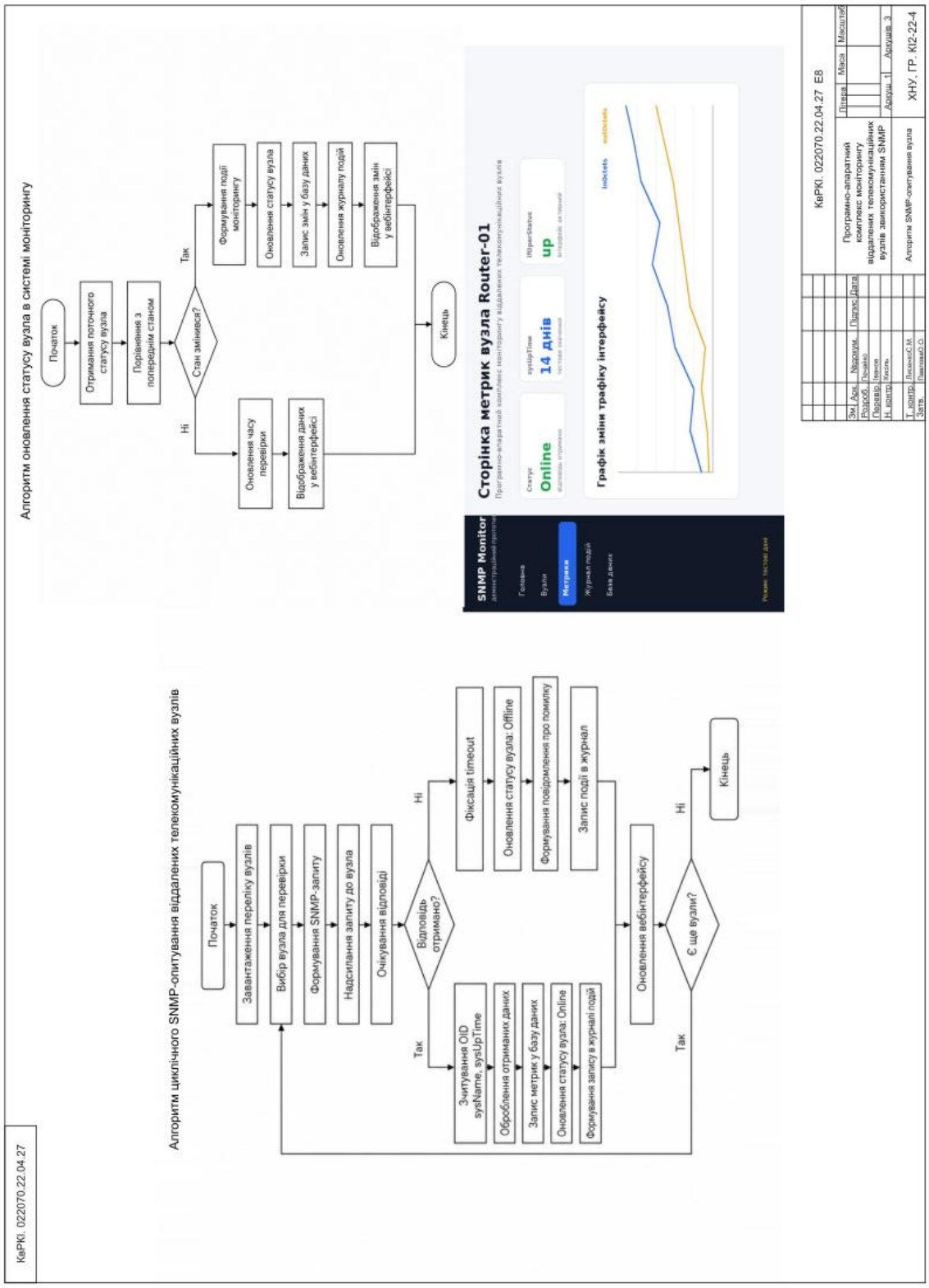
Копія креслення «Архітектура оброблення SNMP-даних»



ДОДАТОК В

(обов’язковий)

Копія креслення «Алгоритм SNMP-опитування вузла»



Зав. кафедри КІІС
д-р. філософії Ользі ПАВЛОВІЙ

Ілля ПЕЧАЙКО
ПІБ здобувача вищої освіти

ФІТ, 4 курсу, групи КІ2-22-4

ЗАЯВА

З правилами чинного Положення про систему забезпечення академічної доброчесності у Хмельницькому національному університеті, згідно з яким виявлення академічного плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту і застосування заходів академічної відповідальності, ознайомлений (а). Про використання спеціалізованих програмних засобів (СПЗ) StrikePlagiarism та Anti-Plagiarism для перевірки кваліфікаційних робіт здобувачів вищої освіти на наявність академічного плагіату оповіщений (а). Надаю університету право на передачу моєї роботи для обробки та збереження в базах даних СПЗ і використання роботи для виявлення академічного плагіату в інших роботах, які перевіряються СПЗ.

Також надаю свою згоду на обробку й збереження університетом моєї роботи в Інституційному репозитарії Хмельницького національного університету.

Робота надається для перевірки в електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

1 травня 2026 року

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

Дипломник: Печайко Ілля Вікторович

Тема: Програмно-апаратний комплекс моніторингу віддалених телекомунікаційних вузлів з використанням SNMP.

Спеціальність: 123 «Комп'ютерна інженерія»

Обсяг кваліфікаційної роботи:

Кількість листів креслень ___3___ Кількість сторінок записки ___89___

1. Короткий зміст роботи та прийнятих рішень: Метою кваліфікаційної роботи є проєктування програмно-апаратного комплексу моніторингу віддалених телекомунікаційних вузлів з використанням SNMP.

2. Висновок про відповідність роботи дипломному завданню: Робота повністю відповідає поставленому завданню.

3. Характеристика виконання кожного розділу, ступінь використання останніх досягнень науки і техніки і передових методів роботи: У першому розділі проаналізовано сучасні методи та програмні засоби моніторингу, розглянуто протокол SNMP і сформовано вимоги до комплексу. У другому розділі спроєктовано структуру системи, розподіл функцій між компонентами та інформаційні потоки. У третьому розділі описано алгоритми роботи, структуру програмного забезпечення, вебінтерфейс та приклади застосування системи.

4. Позитивні сторони роботи: висока практична цінність роботи.

5. Негативні сторони роботи: недостатнє тестування системи з використанням реального мережевого обладнання.

6. Оцінка графічного оформлення та пояснювальної записки роботи: Пояснювальна записка оформлена коректно, згідно діючих стандартів оформлення документації.

7. Відгук про роботу в цілому: Робота виконана на належному науково-технічному рівні.

8. Інші зауваження: _____

9. Оцінка дипломної роботи: "задовільно"

Рецензент (прізвище, ім'я, по батькові, посада, місце роботи) _____

Стецюк Микола Васильович, PhD, ст. викладач кафедри кібербезпеки

"__"_____2026 р. _____(підпис)

РІШЕННЯ ЕКСПЕРТНОЇ КОМІСІЇ

КАФЕДРИ КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА ІНФОРМАЦІЙНИХ СИСТЕМ
ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Назва кваліфікаційної роботи Інформаційна система моніторингу стану здоров'я пацієнтів із оптимізацією планування завдань
Автор Ілля ПЕЧАЙКО
Освітня програма Комп'ютерна інженерія та програмування
Рівень вищої освіти перший (бакалаврський)
Спеціальність 123 Комп'ютерна інженерія
Науковий керівник: канд.фіз.-мат.наук, доц. Олексій ІВАНОВ

На основі аналізу кваліфікаційної роботи на дотримання вимог академічної доброчесності (у т.ч. відсутності ознак академічного плагіату) з урахуванням результатів перевірки роботи спеціалізованим програмним засобом(ами) комісія зробила такий висновок:

№	Висновок	Позначка про відповідність
1	Ознаки академічного плагіату	
1.1	Запозичення, виявлені в роботі, є законними і не є академічним плагіатом (далі – зазначаються підстави віднесення запозичень до правомірних, якщо потрібно). Робота приймається до захисту.	відповідає
1.2	Виявлені запозичення не є академічним плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи (далі зазначаються детальні та аргументовані підстави віднесення запозичень до правомірних). Робота приймається до захисту, але має бути відкоригована.	
1.3	Виявлені запозичення не є академічним плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. Робота може бути допущена до захисту після того як буде відкоригована та доопрацьована і успішно пройде повторну перевірку на академічний плагіат.	
1.4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття текстових запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	
2	Інші види порушень академічної доброчесності	

Підтвердження:

Запозичення, виявлені в роботі, є законними і не є плагіатом, оскільки:

1) усі запозичення фрагментарні, або мають належним чином оформленні посилання;

2) окремі виявлені збіги є загальновживаними фразами або виразами, про що свідчить посилання системи на збіг з джерелами на один фрагмент речення;

3) всі зафіксовані системою ознаки модифікації тексту відносяться до комбінування латиських символів зі україномовними скороченнями індексів в формулах, що не є модифікацією тексту.

4) значна частина знайденого плагіату відноситься до списку використаних джерел

Сумарний обсяг всіх запозичень, визначений системою виявлення збігів/ ідентичності/схожості StrikePlagiarism, складає 2,41%; та системою Anti-Plagiarism складає 1%, що, з урахуванням наведених обгрунтувань, відповідає характеру наукового дослідження і свідчить на користь кваліфікаційної роботи.

01.06.2026

Завідувач кафедри _____ Ольга ПАВЛОВА
Підпис Ім'я, ПРІЗВИЩЕ

Гарант освітньої програми _____ Андрій НІЧЕПОРУК
Підпис Ім'я, ПРІЗВИЩЕ

Керівник кваліфікаційної роботи _____ Олексій ІВАНОВ
Підпис Ім'я, ПРІЗВИЩЕ

Thu May 21 12:01:04 EEST 2026, Медзатий Дмитро Миколайович, Хмельницький національний університет, ХНУ

Anti-Plagiarism (http://ap.km.ua) v-15.701

Максимальне співпадіння з одним документом 1.0%

Словники перевірки: en_US, ru_RU, ua_UA. **Помилок в документах: 9%**

	Документ		Сумарний збіг по Базі Даних	
ID: 271887 Назва: БКР Програмно-апаратний комплекс моніторингу віддалених телекомунікаційних вузлів з використанням SNMP Додано в БД: 2026-05-21 Автора: Ілля ПЕЧАЙКО Керівники: Олексій ІВАНОВ Консультанти: Опоненти:	Символи	Лексеми	Символи	Лексеми
	90892	834	1525 (2%)	18 (2%)

Джерело плагіату

ID	Опис	Наявність плагіату в документі	
		Символи	Лексеми

Протокол аналізу звіту подібності експертом

Заявляю, що я ознайомився (-лась) з Повним звітом подібності, який був згенерований Системою виявлення і запобігання плагіату щодо роботи:

Автор: Ілля ПЕЧАЙКО

Співавтор:

Назва: Програмно-апаратний комплекс моніторингу віддалених телекомунікаційних вузлів з використанням SNMP

Експерт: Олексій ІВАНОВ

Підрозділ: Кафедра комп'ютерної інженерії та інформаційних систем

Коефіцієнт подібності 1:2.41%

Коефіцієнт подібності 2:0.44%

Мікропробіли: 3

Заміна букв: 0

Інтервали: 0

Білі знаки: 2

Дата створення звіту: 2026-05-21 11:13:44.0

Після аналізу Звіту подібності констатую наступне:

☒ **Запозичення, виявлені в роботі є законними і не є плагіатом. Рівень подібності не перевищує допустимої межі. Таким чином робота незалежна і приймається.**

☐ **Запозичення не є плагіатом, але перевищено граничне значення рівня подібностей. Таким чином робота повертається на доопрацювання.**

☐ **Виявлено запозичення і плагіат або навмисні текстові спотворення (маніпуляції), як передбачувані спроби укриття плагіату, які роблять роботу невідповідною вимогам законодавства (Ст. 32. ЗУ Про вищу освіту, пункт 3.1, Ст. 42. ЗУ Про освіту) та вимог НАЗЯВО (Критерій 5), а також кодексу етики і процедурам. Таким чином робота не приймається.**

Обгрунтування:

2026-05-21 Доцент Андрій Нічепорук

Дата експерт