

Центральноукраїнський національний технічний університет
Центр заочної та дистанційної освіти
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
« ____ » _____ 2026 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за першим (бакалаврським) рівнем вищої освіти
на тему
“Програмне забезпечення системи інтелектуального
моніторингу та оптимізації ІТ-систем ЦОД”

Виконав здобувач вищої освіти
IV курсу, групи КН-22з
ОПП «Комп’ютерні науки»
спеціальності 122 «Комп’ютерні науки»
_____ Пасічник Б.С.
« ____ » _____ 2026 р.

Керівник проекту
доктор технічних наук, професор
_____ Смірнов О.А.
« ____ » _____ 2026 р.
Рецензент _____

Центральноукраїнський національний технічний університет
Центр Заочної та дистанційної освіти
Кафедра Кібербезпеки та програмного забезпечення
Освітній ступінь бакалавр
Галузь знань 12 "Інформаційні технології"
Спеціальність 122 "Комп'ютерні науки"
Освітньо-професійна (освітньо-наукова) програма "Комп'ютерні науки"

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____ Олексій СМІРНОВ
« 27 » лютого 2026 року

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ПЕРШИМ (БАКАЛАВРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Пасічнику Богдану Станіславовичу

(прізвище, ім'я, по батькові)

1. Тема роботи *Програмне забезпечення системи інтелектуального моніторингу та оптимізації ІТ-систем ЦОД*

2. Керівник роботи *Смірнов Олексій Анатолійович, докт. техн. наук, професор*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 115-02 від 06.02.2026 року

3. Строк подання студентом роботи до захисту *23.05.2026 р.*

4. Мета та завдання випускної кваліфікаційної роботи: *Метою роботи є розробка програмного забезпечення системи інтелектуального моніторингу та оптимізації ІТ-систем ЦОД*

5. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Призначення та область використання.

2. Перегляд аналогічних існуючих систем.

3. Опис і обґрунтування проектних рішень.

4. Етапи програмування системи.

5. Впровадження системи в промислову експлуатацію.

6. Висновки

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

Структурна схема системи *1 аркуш*

Функціональна схема системи *1 аркуш*

Діаграма процесів *1 аркуш*

Блок-схема алгоритму роботи додатку *2 аркуша*

7. Дата видачі завдання « 27 » лютого 2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.03.2026 р.	
2.	Постановка задачі, оформлення ТЗ	15.03.2026 р.	
3.	Розробка моделі компонента	20.03.2026 р.	
4.	Розробка структур даних	25.03.2026 р.	
5.	Розробка алгоритмів зв'язку та відображення	30.03.2026 р.	
6.	Програмування алгоритмів	10.04.2026 р.	
7.	Оформлення ПЗ	17.04.2026 р.	
8.	Попередній захист роботи	23.05.2026 р.	

Дата видачі завдання
« 27 » лютого 2026 р.

Підпис керівника

Смірнов О.А.
(прізвище та ініціали)

Завдання прийнято до виконання
« 27 » лютого 2026 р.

Підпис здобувача

Пасічник Б.С.
(прізвище та ініціали)

АНОТАЦІЯ

Пасічник Б.С. Програмне забезпечення системи інтелектуального моніторингу та оптимізації ІТ-систем ЦОД. 122 Комп'ютерні науки. Центральноукраїнський національний технічний університет. Кропивницький. 2026.

В даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи інтелектуального моніторингу та оптимізації ІТ-систем ЦОД.

Метою розробки є програмне забезпечення системи інтелектуального моніторингу та оптимізації ІТ-систем ЦОД.

Результат роботи – програмна реалізація системи інтелектуального моніторингу та оптимізації ІТ-систем ЦОД.

В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

Розроблено зручний інтерфейс користувача. Наведені інструкції по роботі з програмними засобами.

Програма може використовуватися на ПЕОМ з ОС Windows 10/11.

Програму розроблено в середовищі Python.

Ключові слова: комп'ютерні науки, інтелектуальний моніторинг та оптимізація, ІТ-система, ЦОД

ABSTRACT

Pasichnyk B.S. Software for the system of intelligent monitoring and optimization of IT systems of data centers. 122 Computer Science. Central Ukrainian National Technical University. Kropyvnytskyi. 2026.

In this final qualification work for the first (bachelor's) level of higher education, software has been developed, which is intended for the system of intelligent monitoring and optimization of IT systems of data centers.

The purpose of the development is the software for the system of intelligent monitoring and optimization of IT systems of data centers.

The result of the work is the software implementation of the system of intelligent monitoring and optimization of IT systems of data centers.

In the process of working on the software model, an analysis of existing hardware and software was performed. All components of the developed software are fully described.

A convenient user interface has been developed. Instructions for working with software are provided.

The program can be used on a PC with OS Windows 10/11.

The program is developed in the Python environment.

Keywords: computer science, intelligent monitoring and optimization, IT system, data center

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ	2
ВСТУП.....	3
1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ	5
1.1 Призначення системи.....	5
1.2 Область застосування.....	6
2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ	9
2.1 Огляд існуючих систем, технологій, архітектур та програмних рішень за профілем теми випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти.....	9
2.2 Обґрунтування вибору засобів для побудови системи та мови програмування.....	25
2.3 Розгорнута постановка завдання	30
3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ	31
3.1 Опис функціонування системи	31
3.2 Розробка структурної схеми.....	35
3.3 Розробка функціональної схеми	40
3.4 Розробка діаграми процесів.....	43
4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ.....	45
4.1 Розробка блок-схем та опис алгоритмів функціонування системи.....	45
4.2 Захист розробленого програмного забезпечення.....	60
5 ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ	62
6 ОСНОВНІ ВИСНОВКИ.....	66
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	68

					ВКРБ-122.26.0060.00.00.ПЗ			
Вим.	Арк.	№ докум.	Підп.	Дата	Програмне забезпечення системи інтелектуального моніторингу та оптимізації ІТ-систем ЦОД	Літ.	Аркуш	Аркушів
Розроб.	Пасічник Б.С.					Б	1	74
Перев.	Смірнов О.А.							
Н.контр.	Коваленко А.С.					ЦНТУ КН-22з		
Затв.	Смірнов О.А.							

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

БД	—	база даних
БЗ	—	база знань
БРС	—	блок розрахункових співвідношень
ВКПК	—	вибір кількості пріоритетних користувачів
ІМА	—	інтелектуальний мережний адміністратор
КОМ	—	корпоративні обчислювальні мережі
ЛОМ	—	локальні обчислювальні мережі
МА	—	мережне адміністрування
ПЗ	—	програмне забезпечення
СУБД	—	система управління базою даних
ЦОД	—	центр обробки даних
BMS	—	Building Management System
CMIP	—	протокол загальної керуючої інформації
DCIM	—	Data Center Infrastructure Management
DHCP	—	протокол динамічної конфігурації вузла
FMP	—	HPOpenView Fault Management Platform
MIB	—	база даних інформації управління
OEMF	—	HPOpenView Element Management Framework
OER	—	Optimized Edge Routing
OV	—	HPOpenView
Pf	—	Performance Routing
RMON	—	протокол моніторингу комп'ютерних мереж
SNMP	—	простий протокол управління мережею
WINS	—	служба зіставлення netbios-імен комп'ютерів з ір-адресами вузлів

ВСТУП

Актуальність теми. Щоб знизити витрати й оптимізувати управління багатоскладовою структурою центра обробки даних, моніторинг ІТ-інфраструктури необхідно здійснювати в цілодобовому режимі. Сьогодні в розпорядженні операторів центрів обробки даних (ЦОД) є безліч корисних інструментів для моніторингу й оптимізації ІТ-систем і допоміжного устаткування, а також для управління ними.

За останні кілька років центри обробки даних стали найважливішим елементом корпоративної ІТ-інфраструктури. Сучасний ЦОД – це сукупність ІТ-устаткування й різних інженерних систем. Останні забезпечують кондиціонування повітря, безперебійне й гарантоване енергопостачання, контроль, управління доступом і т.д. У ЦОД найчастіше зберігаються критично важливі дані, і навіть тимчасовий збій у край негативно позначається на роботі підприємства. Саме тому дороге устаткування повинне бути надійно захищене від несанкціонованого проникнення, пожежі й інших непередбачених ситуацій.

Протягом життєвого циклу ЦОД інженерне й ІТ-устаткування постійно піддається модернізації, внаслідок чого виникають різні неузгодженості, які надалі можуть привести до його некоректної роботи. Збої відбуваються з різних причин: через розміщення апаратних засобів без обліку їхніх характеристик, недотримання температурного режиму й навіть переміщення устаткування на нове місце (перерозподіл потоків повітря може привести до перебоїв у функціонуванні).

Для виправлення ситуації необхідно регулярно аналізувати інформацію, що надходить із датчиків, і періодично виконувати аудит всіх систем – у середньостроковій перспективі це дозволяє скорегувати якість роботи ЦОД.

Не менш важливе питання енергоефективності, тому що електроенергія – основний ресурс, що споживає ЦОД. Щоб знизити витрати й оптимізувати,

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		3

управління багатоскладовою структурою, моніторинг ІТ-інфраструктури повинен здійснюватися в цілодобовому режимі. Сьогодні в розпорядженні операторів ЦОД є безліч корисних інструментів для моніторингу й оптимізації ІТ-систем і допоміжного устаткування, а також для управління ними.

Мета й завдання дослідження. Метою роботи є програмне забезпечення системи інтелектуального моніторингу та оптимізації ІТ-систем ЦОД.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

- Огляд існуючих систем інтелектуального моніторингу та оптимізації ІТ-систем ЦОД.
- Дослідження системи інтелектуального моніторингу та оптимізації ІТ-систем ЦОД.
- Програмна реалізація системи інтелектуального моніторингу та оптимізації ІТ-систем ЦОД.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі інтелектуального моніторингу та оптимізації ІТ-систем ЦОД.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи інтелектуального моніторингу та оптимізації ІТ-систем ЦОД, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Щоб підтримувати ЦОД в «доброму здоров'ї», деякі компанії розробляють власне програмне забезпечення (ПЗ), що допомагає відслідковувати роботу устаткування. На створення й підтримку таких рішень, які звичайно розраховані на задоволення вузькоспеціальних вимог конкретного ЦОД, потрібні чималі внутрішні ресурси.

Ще один спосіб автоматизації контролю роботи устаткування – застосування системи управління будинком (Building Management System, BMS), або автоматизованої системи управління будинком (АСУБ). Локальні контролери поєднуються в єдину мережу, що перебуває під управлінням центрального комп'ютера, на якому встановлений комплекс необхідного програмного забезпечення.

Одержуючи сигнали від датчиків виміру температури, вологості, протікань, струму й інших параметрів, контролери відправляють необхідні команди диспетчерові або виконавчим пристроям (залежно від обставин) і тим самим допомагають здійснювати управління основними інженерними системами будинку: вентиляцією, охолодженням, газо- і водопостачанням, електроживленням, висвітленням і безпекою.

Таким чином, BMS дозволяє організувати управління інженерним устаткуванням і його моніторинг. Крім того, з її допомогою власники будинків можуть виконувати диспетчеризацію інженерних систем, відслідковуючи навіть мінімальні порушення й відхилення від норми, причому BMS інформує про можливі проблеми й небезпеки. Крім цього, система збирає статистичні дані, які допомагають аналізувати ступінь ефективності роботи устаткування й набудовувати його належним чином.

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		5

1.2 Область застосування

На ринку є рішення, які допомагають повністю оптимізувати управління характеристиками й життєвим циклом ЦОД, – це системи моніторингу й управління ЦОД (Data Center Infrastructure Management, DCIM).

Насамперед DCIM орієнтована на підвищення економічної ефективності: з її допомогою організації можуть оптимізувати енергоспоживання всієї своєї інфраструктури й повністю управляти їм. Коефіцієнт енергоефективності PUE знижується, і, відповідно, зменшуються витрати на експлуатацію ЦОД.

Однак пряма економія – не єдина перевага DCIM. У широкому змісті ці системи допомагають підприємствам домагатися рішення своїх стратегічних завдань. Програмне забезпечення DCIM відслідковує, вимірює, реєструє інженерно-технічні ресурси ЦОД, управляє ними, оптимізує енергоспоживання, надає аналітику по системах і процесам.

У складі програмного забезпечення DCIM присутні різні функціональні модулі, що дозволяє впроваджувати систему поетапно: наприклад, спочатку увести до ладу базовий модуль, а потім, залежно від пріоритетів департаменту, інші:

- Базовий модуль відповідає за збір інформації.
- Модулі Energy і Capacity – за її аналіз.
- Capacity і Asset – за планування, експлуатацію.
- Work Order і Asset Inspection – за ведення статистики, інспектування встановленого устаткування, управління персоналом і різними завданнями.

DCIM надає необхідну інформацію кожному відділу обслуговування ЦОД:

- начальник ІТ-служби може контролювати, де розташовується конкретний сервер і як нові сервіси впливають на наявну систему живлення й охолодження ЦОД, а також на завантаження серверів;

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		6

– начальник інженерно-технічної служби одержує можливість виявити зони температурних максимумів, причини подачі аварійних сигналів, оцінити значення PUE, оптимізувати розташування устаткування;

– фахівець, відповідальний за управління ЦОД, зможе розрахувати витрати на експлуатацію й ефективність ЦОД, спрогнозувати потребу в збільшенні потужності існуючого ЦОД або будівництві нового.

DCIM дозволяє видавати в зручному виді необхідний зріз інформації для керівників будь-якого рівня й різних департаментів. Програмний комплекс забезпечує можливість аналізу поточної ситуації в ЦОД, перевірки його експлуатації й роботи устаткування, виявлення проблемних місць, а також надання рекомендацій керуючому персоналу. Впровадження DCIM дозволяє вирішити наступні завдання:

- виявлення й усунення причин перебоїв у роботі устаткування;
- зниження рівня споживання електроенергії;
- збільшення віддачі від капіталовкладень і скорочення чисельності персоналу;
- централізація формування замовлень устаткування й запчастин;
- централізація завдань на обслуговування, аудита, ремонт устаткування;
- планування модернізації й росту потужності ЦОД;
- аналіз розташовуваної ємності стійок, мережі, запасу по навантажувальній здатності, запасу по енерго/холодопостачанню;
- планування заповнення стійок;
- синхронізація циклів модернізації ІТ-інфраструктури, інженерної інфраструктури й часу життя ЦОД.

Очевидно, що безперервний моніторинг інфраструктури ЦОД, своєчасне проведення всіх необхідних робіт з її оптимізації й раціональне управління допоміжним устаткуванням – застава ефективної й безперебійної роботи критично важливої обчислювальної інфраструктури. Головна перевага DCIM у

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		7

тім, що вона орієнтована саме на рішення завдань експлуатації ЦОД і дозволяє управляти ІТ- і інженерним устаткуванням, персоналом.

Крім того, на відміну від BMS, система моніторингу й управління центром обробки даних здатна проорокувати виникнення проблем завдяки програмним модулям аналітики. У результаті DCIM не тільки поліпшує управління потужностями й активами, але й підвищує загальну продуктивність, а також збільшує тривалість життєвого циклу ЦОД.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи інтелектуального моніторингу та оптимізації ІТ-систем ЦОД, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

КБПЗ-2026

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		8

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

2.1 Огляд існуючих систем, технологій, архітектур, програмних рішень за профілем теми випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти

Незапланований простій сервера може призвести до серйозних збоїв у роботі веб-сайтів та користувачів без належного програмного забезпечення для моніторингу сервера. Одне непомічене вузьке місце може спричинити повільне завантаження, невдалі транзакції або повні збої. Правильні інструменти моніторингу сервера допомагають виявляти проблеми в режимі реального часу та пропонують проактивний підхід до управління інфраструктурою. Ці рішення підтримують безперебійну роботу, оптимізують використання ресурсів та виявляють інциденти на ранній стадії у фізичних та віртуальних середовищах, а також у хмарних та контейнерних системах. У цьому розділі розглядаються найкращі інструменти моніторингу серверів на 2026 рік та надаються практичні поради щодо вибору найкращого варіанту.

Netdata

Netdata працює як платформа для моніторингу інфраструктури в режимі реального часу з відкритим кодом, побудована на розподіленій архітектурі. Програмне забезпечення збирає дані з інтервалом в одну секунду, а найгірший випадок – затримка менше двох секунд від події до візуалізації в будь-якому масштабі. Така секундна деталізація фіксує операційні аномалії, що тривають менше 10 секунд, що пропускають традиційні системи моніторингу, що працюють з інтервалом 10-60 секунд.

Легкий обсяг платформи використовує приблизно 5% одного ядра процесора та 150 МБ оперативної пам'яті на виробничих системах. Дослідження Амстердамського університету оцінює Netdata як найбільш енергоефективний

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		9

інструмент для моніторингу систем на базі Docker, перевершуючи інші рішення для моніторингу за використанням процесора, використанням оперативної пам'яті та часом виконання. Споживання ресурсів ще більше падає, коли машинне навчання та сповіщення вимкнено.

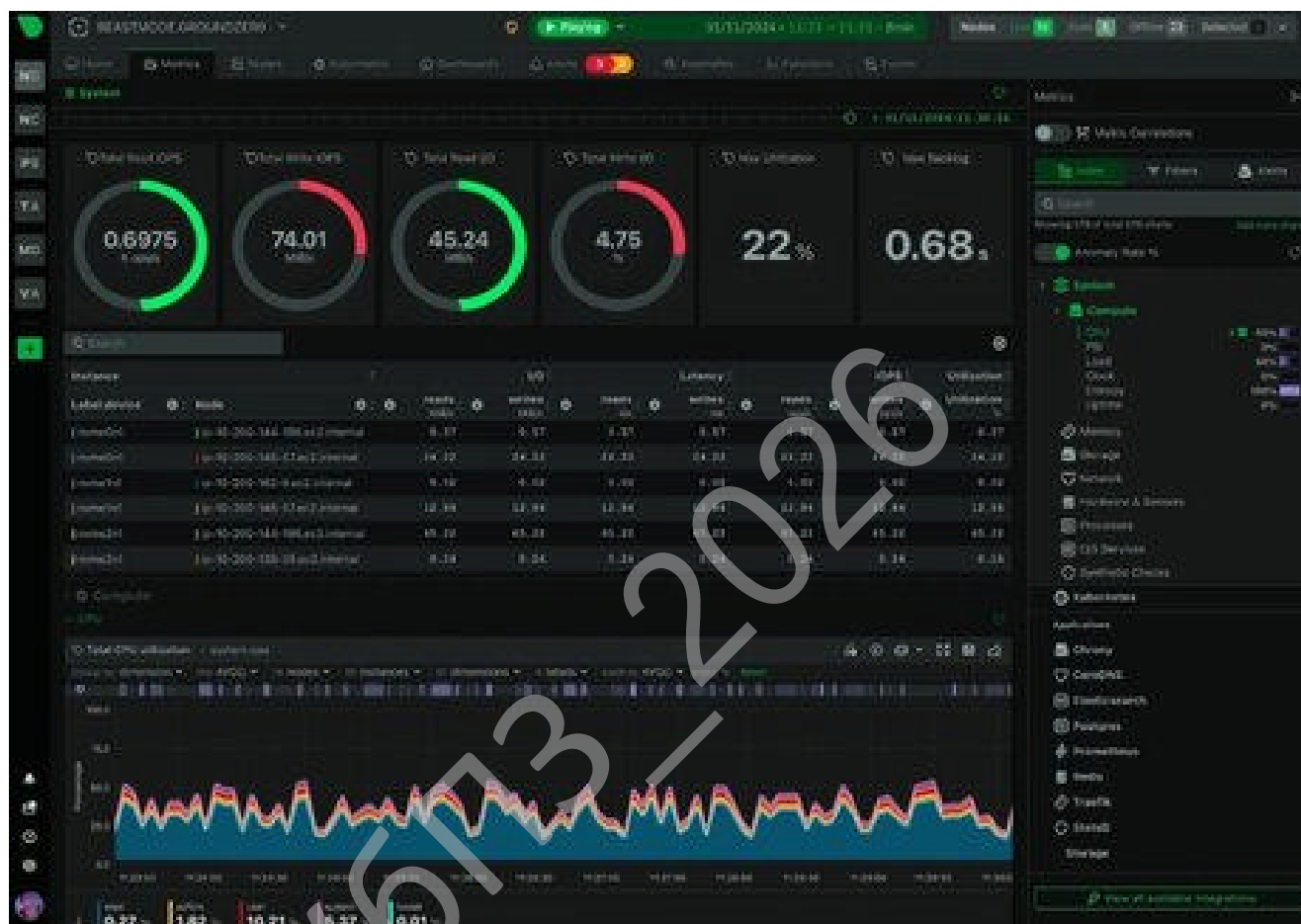


Рисунок 2.1 – Інтерфейс користувача netdata.cloud

Підхід Netdata з нульовою конфігурацією робить можливим негайне розгортання без складних процесів налаштування. Система автоматично виявляє та контролює понад 800 інтеграцій "з коробки", охоплюючи системні ресурси, сховище, мережі, контейнери, віртуальні машини та хмарну інфраструктуру. Інсталяція вимагає лише однієї команди, а повна панель інструментів доступна протягом кількох хвилин.

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		10

Платформа включає усунення несправностей на базі штучного інтелекту завдяки інтегрованим автоматизації. Anomaly Advisor використовує 18 консенсусних моделей машинного навчання для кожної метрики, щоб зменшити кількість хибнопозитивних результатів та виявляти проблеми на ранній стадії. Механізм аналізу першопричин миттєво визначає поширення збоїв, а звітність за допомогою штучного інтелекту вказує на точне джерело проблем. Ці можливості ставить Netdata серед найкращих програмних засобів для моніторингу серверів, доступних сьогодні.

Його розподілена архітектура усуває централізовані вузькі місця, дозволяючи кожному вузлу працювати незалежно зі стабільною затримкою менше 2 секунд, незалежно від масштабу. Безпека залишається високою завдяки сертифікації SOC 2 Type 2, локальному сховищу метрик та вбудованій інтеграції зі Slack, PagerDuty, Microsoft Teams та іншими платформами сповіщень.

Datadog

Datadog постачає хмарне програмне забезпечення для моніторингу серверів за моделлю «Програмне забезпечення як послуга». Воно забезпечує видимість програм, інфраструктури та журналів з єдиної платформи. Інструмент поєднує метрики, трасування та журнали, щоб створити повне уявлення про продуктивність системи, дозволяючи командам пов'язувати дані на різних рівнях свого технологічного стеку.

Можливості негайного оповіщення повідомляють адміністраторів про проблеми з продуктивністю сервера в міру їх виникнення. Карти хостів і контейнерів візуалізують стан серверів або контейнерів в одному інтерфейсі, тоді як карти служб відображають потоки даних програм і залежності в режимі реального часу. Синхронізовані панелі інструментів відстежують інциденти за різними показниками за допомогою спільної структури тегів, що допомагає командам швидше виявляти та вирішувати проблеми.

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		11



Рисунок 2.2 – Інтерфейс користувача Datadog

Datadog підтримує моніторинг на серверах типу «голе метал», віртуальних машинах та в приватних хмарних середовищах, розміщених у внутрішніх центрах обробки даних. Він інтегрується з корпоративними платформами, такими як IBM WebSphere та VMware vSphere, для моніторингу віртуалізованих програм. Організації, що використовують системи Unix, включаючи IBM AIX, можуть розгорнути власний агент для збору системних показників. Інструменти керування конфігурацією, такі як Puppet, інтегруються з Datadog та дозволяють командам відстежувати зміни інфраструктури разом з автоматизованим розгортанням.

Агент Datadog збирає метрики з хостів і контейнерів з точністю до 15 секунд. Готові інтеграції з більш ніж 500 технологіями забезпечують охоплення в різних середовищах. Пошук і аналітика на основі тегів дозволяють командам створювати точні сповіщення та зосереджувати розслідування на певних компонентах інфраструктури, тоді як машинне навчання автоматично виявляє аномальну поведінку.

Створення інформаційних панелей здійснюється на основі шаблонів із попередньо встановленими опціями для більшості сервісів. Команди можуть налаштовувати віджети, визначати порогові значення сповіщень та отримувати сповіщення електронною поштою або через платформи для співпраці. Мобільні додатки для iOS та Android надають push-сповіщення та дозволяють адміністраторам контролювати продуктивність інфраструктури з будь-якого місця.

New Relic

New Relic забезпечує повноцінне спостереження за допомогою моніторингу інфраструктури, вбудованого безпосередньо в відстеження продуктивності додатків. Платформа поєднує дані інфраструктури з метриками APM та пропонує видимість усього технологічного стеку без перемикання між окремими інструментами. Цей уніфікований підхід дозволяє командам вирішувати проблеми з продуктивністю до 80% швидше, пов'язуючи стан інфраструктури з контекстом додатків та змінами конфігурації.

Агент інфраструктури збирає дані про продуктивність та стан системних ресурсів і процесів у системах Linux, macOS та Windows. Метод інструкцій з встановлення супроводжує команди під час розгортання та автоматично виявляє сумісність системи, при цьому більшість налаштувань виконується менш ніж за 15 хвилин. Агент підтримує локальні сервери та віртуальні машини, а локальні інтеграції розширюють моніторинг на бази даних, служби обміну повідомленнями та сервери додатків.

Хмарні інтеграції працюють без встановлення агента чи додаткової інфраструктури. Команди просто підключають свій обліковий запис New Relic до Amazon Web Services, Microsoft Azure або Google Cloud Platform, щоб розпочати збір даних. Середовища Kubernetes отримують видимість завдяки підключенню облікових записів, що дозволяє моніторити кластер, надсилати сповіщення та відстежувати продуктивність з централізованого інтерфейсу.

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		13



Рисунок 2.3 – Інтерфейс користувача New Relic

На сторінці «Хости» відображаються дані про продуктивність систем, мереж, процесів і сховищ в одному місці. Динамічні діаграми показують показники хостів і програм разом, що спрощує зв'язок падіння продуктивності зі змінами в інфраструктурі. Сторінка «Інвентаризація» дозволяє командам шукати на хостах, щоб ідентифікувати встановлені пакети, конфігурації або служби.

Метрики в реальному часі та відстеження змін скорочують час вирішення проблем, пов'язуючи проблеми продуктивності із системними подіями. Журнали в контексті поєднують дані інфраструктури з журналами програм для швидшого усунення несправностей. Виявлення аномалій на основі машинного навчання автоматично визначає аномальну поведінку, а карти топології та інтелектуальні сповіщення допомагають командам розуміти залежності, візуалізувати вплив та швидше вирішувати проблеми.

Dotcom-Monitor

Dotcom-Monitor підходить до програмного забезпечення для моніторингу серверів через синтетичний моніторинг реального браузера, який тестує функціональність, а не просто доступність. Платформа покроково обходить критичні робочі процеси, такі як форми входу, транзакції електронної комерції та динамічний контент, використовуючи реальні браузери, щоб виявляти збої до того, як користувачі зіткнуться з ними. З 1998 року компанія допомогла понад 10 000 організацій підтримувати продуктивність веб-сайтів, безперебійну роботу та надійність програм.

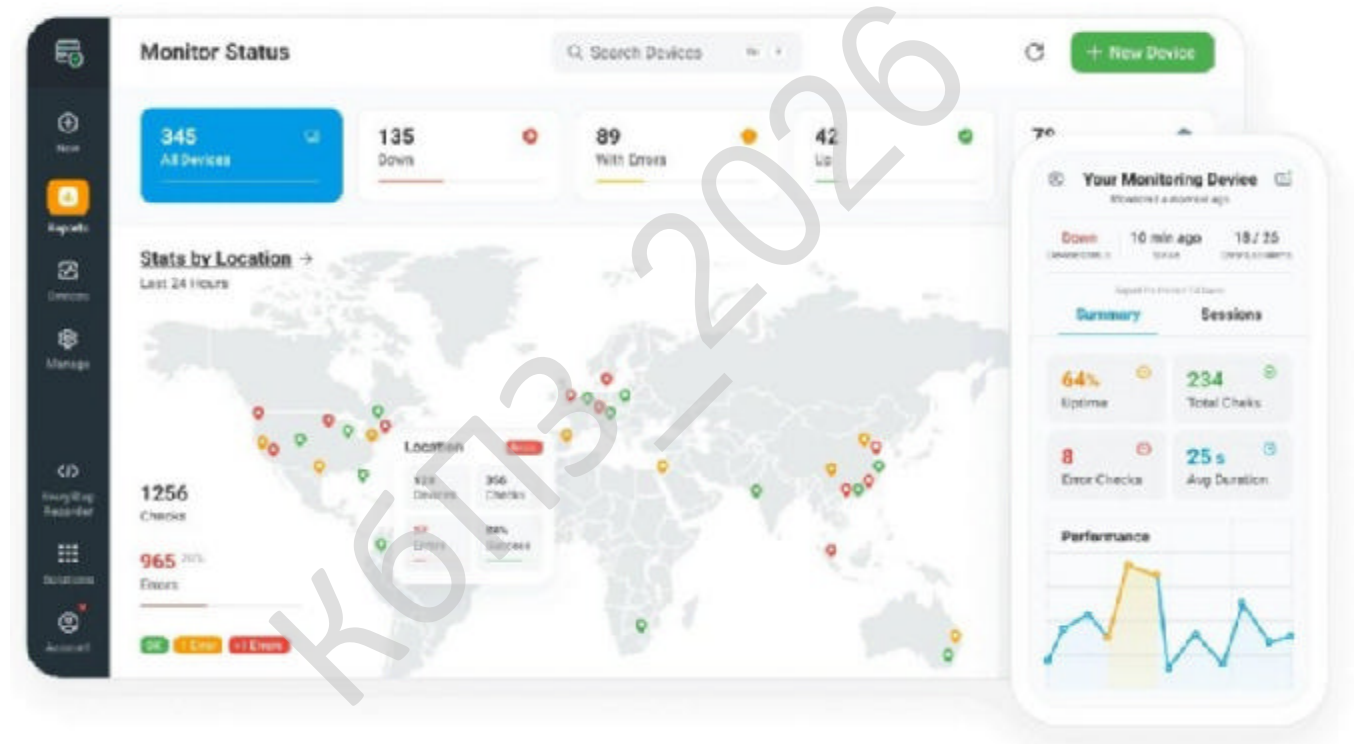


Рисунок 2.4 – Інтерфейс користувача Dotcom-Monitor

Кілька спеціалізованих платформ моніторингу працюють в одному уніфікованому інтерфейсі. ServerView перевіряє час безперебійної роботи та продуктивність серверів, мереж і пристроїв за допомогою постійного моніторингу інфраструктури від глобальних агентів. WebView виконує

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		15

тестування веб-сервісів для HTTP/S, SOAP та REST API. UserView імітує реальну взаємодію користувачів за допомогою сценаріїв робочих процесів браузера, тоді як BrowserView відстежує швидкість завантаження сторінок та час їх використання. MetricsView контролює лічильники продуктивності обладнання, такі як завантаження процесора, використання пам'яті та обсяг дискового простору. Команди можуть налаштовувати моніторинг веб-сайтів, API, поштових серверів, FTP-сервісів, потокового медіа, VoIP-систем, DNS-серверів та мережевих протоколів. EveryStep Web Recorder забезпечує запис сценаріїв одним клацанням миші, що фіксує взаємодію з браузером без ручного кодування. Скрипти відтворюються в реальних браузерах та взаємодіють із програмами точно так само, як і кінцеві користувачі. Інтервали моніторингу коливаються від однієї хвилини до трьох годин залежно від вимог.

Понад 30 глобальних точок моніторингу забезпечують послідовне відстеження продуктивності з різних регіонів. Сповіщення про тривоги передаються електронною поштою, SMS, Slack, PagerDuty та іншими інтеграціями, а сповіщення містять діагностичні дані, такі як трасування маршрутів та знімки екрана. Налаштовувані інформаційні панелі відображають час безвідмовної роботи, час відгуку та тенденції продуктивності, а звіти генеруються в різних форматах. Великі організації, включаючи Comcast, Dell та Volvo, покладаються на Dotcom-Monitor для підтримки надійності інфраструктури та забезпечення постійної доступності послуг.

LogicMonitor

LogicMonitor поєднує архітектуру платформи SaaS з легкими безагентними колекторами, які розгортаються безпосередньо в мережах клієнтів для поєднання хмарних інтерфейсів з локальною інфраструктурою. Цей гібридний підхід усуває традиційні компроміси між хмарним моніторингом та локальним збором даних. Організації отримують повну видимість у центрах обробки даних, хмарних середовищах та розподілених місцях завдяки єдиному інтерфейсу.

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		16

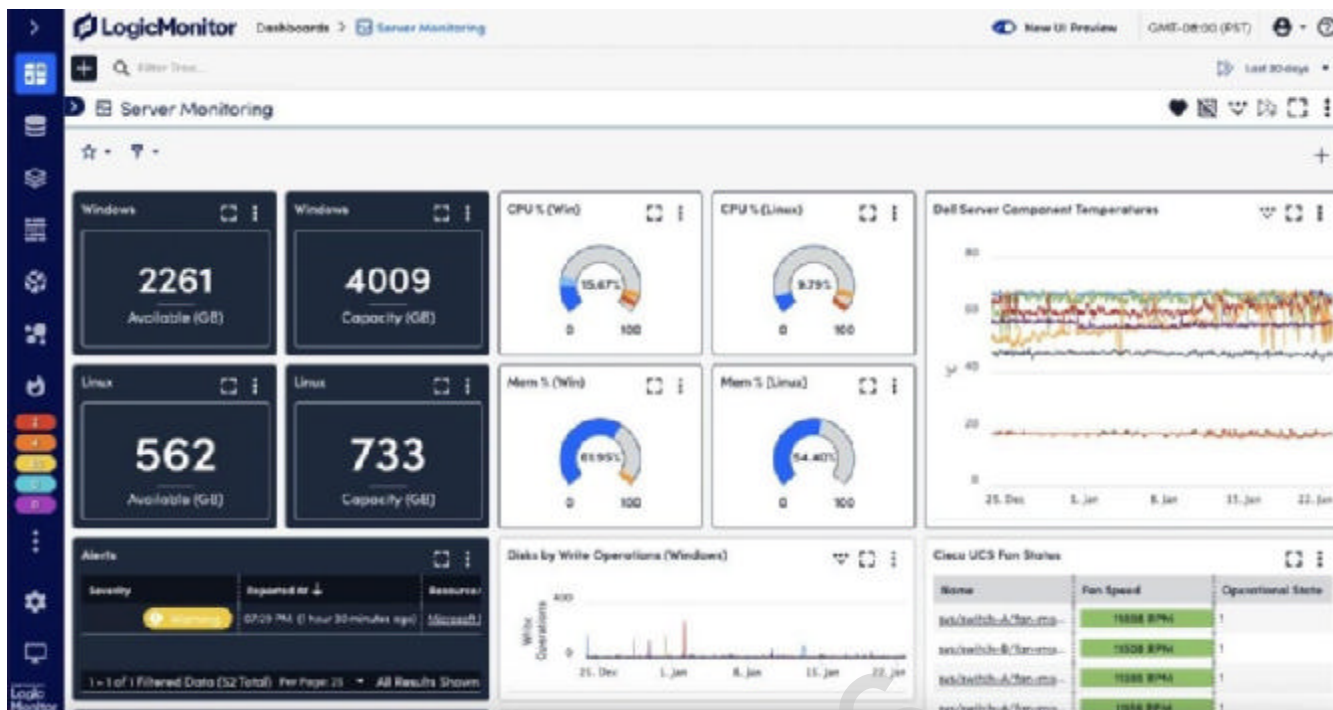


Рисунок 2.5 – Інтерфейс користувача LogicMonitor

Безагентні колектори платформи автоматично виявляють цілі інфраструктури протягом кількох хвилин завдяки інтелектуальним можливостям сканування. Netscan ідентифікує ресурси в межах визначених діапазонів IP-адрес, що спрощує підключення пристроїв без ручного налаштування. LogicMonitor підтримує моніторинг за допомогою таких протоколів, як API, SNMP, NetFlow, WinRM, PowerShell та SSH. Сумісність з різними постачальниками включає обладнання від Cisco, Dell Technologies, VMware, Palo Alto Networks, Juniper Networks та Citrix з автоматичним налаштуванням для швидкого розгортання.

Edwin AI виконує роль двигуна AIOps платформи та застосовує машинне навчання без нагляду до операційних даних. Агент AI зменшує шум сповіщень завдяки інтелектуальному сортуванню та пришвидшує вирішення проблем за допомогою автоматизованих наборів процедур. Він зберігає інституційні знання, виявляючи перевірені рішення з минулих інцидентів. Організації повідомляють про значні покращення, включаючи до 90% менше шуму сповіщень та швидший час реагування на інциденти.

Можливості мережевого моніторингу забезпечують повний огляд маршрутизаторів, комутаторів, брандмауерів, балансувальників навантаження та середовищ SD-WAN. Платформа автоматично створює топологічні карти з моніторингом з урахуванням залежностей та динамічно налаштовується в міру розвитку інфраструктури. Інженери можуть співвідносити журнали, показники продуктивності та події в одному місці для швидшої діагностики проблем.

Інтелектуальні сповіщення замінюють статичні порогові значення динамічними базовими рівнями, що базуються на історичних тенденціях. Попередньо вбудовані інтеграції з системами видачі заявок та інформаційними панелями забезпечують швидше вирішення інцидентів. Розгортання на основі колекторів дозволяє командам швидко вводити в експлуатацію, забезпечуючи вимірні покращення операційної ефективності та скорочення середнього часу вирішення.

Zabbix

Zabbix – це досвідчене рішення для моніторингу з відкритим кодом, якому довіряють понад 300 000 установок по всьому світу, включаючи 54 компанії зі списку Fortune 500, такі як Salesforce, NEC, Orange та Airbus. Платформа позиціонує себе як універсальне рішення для спостереження як для ІТ-середовищ, так і для ОТ-середовищ. Організації можуть розгорнути її локально як безкоштовне програмне забезпечення або використовувати керований хмарний хостинг через Zabbix Cloud.

Zabbix випускається за ліцензією GNU Affero General Public License версії 3 та не вимагає ліцензійних платежів для доступу до повного набору функцій. Організації отримують повний контроль, розгортаючи програмне забезпечення на власній інфраструктурі без жодних витрат. Як альтернатива, Zabbix Cloud займається хостингом, масштабуванням та безпекою ззовні. Ця модель з відкритим кодом забезпечує можливості моніторингу корпоративного рівня без періодичних витрат на підписку, типових для комерційних рішень.

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		18



Рисунок 2.6 – Інтерфейс користувача Zabbix

Платформа моніторить мережі, сервери, віртуальні машини, хмарні сервіси, контейнери, пристрої Інтернету речей та сервери додатків. Автоматичне виявлення ідентифікує мережеві пристрої та системні ресурси за допомогою вбудованих інтеграцій. Збір метрик підтримує як агентні, так і безагентні методи, збираючи дані про продуктивність з операційних систем, платформ віртуалізації, таких як VMware, контейнерних платформ, таких як Docker та Kubernetes, баз даних та API.

Zabbix підтримує широкий спектр протоколів, включаючи SNMP, IPMI, JMX, SSH, ODBC та Prometheus, що дозволяє моніторити обладнання, програми та інфраструктуру. Виявлення проблем у реальному часі дозволяє швидко співвідносити вхідні проблеми та визначати їх першопричини. Багатокористувацька оренда дозволяє командам контролювати кілька середовищ з одного інтерфейсу.

Можливості інтеграції включають вебхуки для ServiceNow та Jira, REST API для автоматизації та сумісність з Grafana для візуалізації. Завдяки сотням доступних шаблонів та інтеграцій, Zabbix забезпечує гнучкий та масштабований моніторинг для складних інфраструктурних середовищ.

Із числа провідних постачальників рішень DCIM в Україні давно й успішно працюють Schneider Electric, Emerson Network Power, Rittal та Raritan.

Panduit

Зате дуже активні в Україні компанії, які історично були відомі як постачальники СКС і відповідних систем керування. Із цієї категорії постачальників одне з найбільш функціональних рішень пропонує компанія Panduit. Крім автоматичного документування всіх відключень і підключень і інших функцій «інтелектуальних» СКС, її продукт PIM підтримує роботу самих різних датчиків і багато які інші з описаних вище функцій DCIM. Одними з найбільш важливих напрямків його розвитку називається вдосконалювання функцій по прогнозуванню тенденцій зміни навантаження з видачею рекомендацій з необхідних дій (на даний момент ці функції реалізовані лише частково), забезпечення динамічного розрахунку PUE, а також агрегації як можна більшої кількості параметрів у єдину систему для виявлення джерел проблем (Root Cause Analysis) і, відповідно, мінімізації часу простою.

TE Connectivity

Компанія TE Connectivity пропонує дві системи автоматизованого керування кабельною інфраструктурою ЦОД: AMPTRAC і Quareo. У системі AMPTRAC моніторинг перемикань на фізичному рівні реалізується за допомогою додаткового провідника в комутаційних шнурах і сенсорних накладках для комутаційних панелей. У системі Quareo використовується технологія Connection Point Identification (CPID) з мікрочипами, убудованими в качани комутаційних шнурів, і зчитувачами в комутаційних панелях. Обидві системи працюють під керуванням ПЗ Infrastructure Configuration Manager (ICM). До аналізаторів системи AMPTRAC і контролерам комутаційних панелей Quareo

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		20

можуть підключатися датчики температури. ПЗ ICM інтегрується із системами DCIM за допомогою API і SDK, надаючи замовникові (адміністраторові) достовірну інформацію про з'єднання на фізичному рівні. Він підкреслює, що оскільки історично рішення AMPTRAC розвивалося разом з ПЗ управління компанії iTracs, те й з її системою DCIM інтеграція реалізується без яких-небудь додаткових доробок. Експерти TE Connectivity не розглядають ICM як систему класу DCIM, думаючи, що вона стане в ЦОД сполучною ланкою між інженерними системами й IT. У роботі перебувають кілька проектів по інтеграції ICM с системами DCIM Emerson/Avocent, iTracs і Schneider Electric.

Провідна роль у розробці й реалізації DCIM приділяється тим виробникам, які вносять найбільший вклад у програмне й апаратне забезпечення ЦОД, а системи сторонніх виробників повинні «бесшовно» інтегруватися в DCIM, доповнюючи базовий керуючий комплекс необхідними функціями. DCIM буде розвиватися в напрямку стандартизації програмних і апаратних інтерфейсів для забезпечення взаємодії як можна більшого числа систем контролю за критично важливими параметрами інженерної інфраструктури ЦОД і встановлених у ньому IT-систем.

Ri Technologies, ще одна історично кабельна компанія, пропонує нове рішення CenterMind, що складається із трьох підсистем, об'єднаних єдиним ПЗ. Підсистема Ri PatchView контролює кабельну й мережну інфраструктури, CenterMind G+ – параметри робітничого середовища усередині шафи (для цього використовуються різні датчики: температури, вологості, протікання води, відкриття дверей), а CenterMind P+ – споживання електроенергії (за допомогою інтелектуальних PDU). Кожна із цих підсистем може функціонувати як разом з іншими, так і окремо, «закриваючи» свою зону відповідальності.

Environment Management System

Із числа компаній, які для багатьох асоціюються з ДПЖ, убік DCIM рухаються вже згадувані Eaton і Tripp Lite, а також компанія Delta Electronics. Delta у співробітництві із провідними виробниками серверів і систем зберігання

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		21

даних у цей час завершує роботу над застосунками, що дозволяють управляти ІТ-інфраструктурою. Що стосується керування інженерною інфраструктурою й робітничим середовищем ЦОД, рішення компанії забезпечують «практично 100-процентний контроль» за ними.

Система Environment Management System (EMS) дозволяє відслідковувати різні параметри, такі як температура й вологість, фіксувати влучення води, виникнення вогню й диму, несанкціоноване проникнення. Вона складається з ПЗ керування InsightPower Manager, центральної станції контролю й моніторингу EnviroStation і датчиків навколишнього середовища Enviroprobe.

Ще однією групою компаній, що реалізує гасла, написані на прапорі DCIM, є виробники перемикачів KVM, до яких відносяться Raritan і Aten. Остання, наприклад, пропонує систему для контролю за станом робітничого середовища й керування електроживленням. Крім базового моніторингу, вона здатна проводити розрахунок енергоефективності. Використовуючи дані, реєструємі блоками PDU і датчиками стану робітничого середовища, програма Eco Sensors по зонах розраховує індекс охолодження стійки Rack Cooling Index (RCI) і індекс температури зворотного повітряного потоку Return Temperature Index (RTI). Таким чином, вона надає відомості для діагностики ефективності споживання й аналізу можливостей економії електроенергії.

Aegis DCIM

Компанія Conteg пропонує систему Aegis DCIM для «комплексного моніторингу й керування всією інфраструктурою ЦОД». Раніше ця компанія була в основному відома як постачальник шаф, але в останні роки намагається закріпитися на ринку комплексних інфраструктурних рішень для ЦОД. Із широкого функціонала Aegis відзначаю розрахунок показників PUE, EUE і EER з візуалізацією результатів на панелі оператора, створення звітів про енергоспоживання ІТ-устаткування на рівні всього ЦОД, ряду шаф, окремої шафи або окремої розетки. Багато кінцевих користувачів сприймають DCIM як систему моніторингу з якимось розширеним функціоналом. Але ідея DCIM істотно

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		22

ширше, вона охоплює не просто моніторинг і контроль на рівні встаткування, а оптимізацію, керування, підвищення ефективності на більше глобальному рівні – на рівні ЦОД або групи розподілених ЦОД у цілому.

Ще одна компанія – постачальник монтажних конструктивів Shtugger – повідомила нас про наявність у її портфелі пропозицій відразу двох рішень з контролю стану робітничого середовища в 19-дюймових шафах. Одне з них побудовано на базі iPDU – блоків мережних розеток з додатковим функціоналом вилученого контролю й керування живленням, а також моніторингу стану робітничого середовища в серверній або телекомунікаційній шафі. До одного блоку iPDU можливе підключення до трьох датчиків температури/вологості (верх, середина й низ шафи) або системи кондиціонування й вентиляторних модулів (стельові й 19-дюймові) з убудованим електронним термостатом. При досягненні критичної оцінки системи охолодження можуть включатися/вимикатися автоматично.

Як вибрати правильний інструмент для моніторингу сервера

Вибір правильного інструменту моніторингу серверів вимагає структурованого процесу оцінки, який починається з чіткого визначення бізнес-та технічних вимог. Тип інфраструктури, пріоритети продуктивності та бюджетні обмеження впливають на рішення. Організації повинні спочатку визначити, де працюють сервери – локально, у хмарі чи в гібридних середовищах – оскільки кожна конфігурація вимагає різних можливостей моніторингу. Локальні середовища вимагають глибокого контролю апаратного забезпечення для відстеження системних компонентів, процесів та продуктивності внутрішньої мережі. Хмарні середовища вимагають безперешкодної інтеграції з такими платформами, як AWS, Azure або Google Cloud, а також можливості динамічного масштабування. Рішення для моніторингу також повинні підтримувати сучасні компоненти інфраструктури, включаючи віртуальні машини, контейнери та мікросервіси. Багато організацій також використовують відновлені сервери як частину своєї інфраструктурної стратегії для зниження витрат та продовження

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		23

життєвого циклу обладнання. Компанії, які покладаються на послуги з відновлення електроніки, часто переносять відновлені сервери, мережеві пристрої та системи зберігання даних у виробничі середовища. У цих випадках інструменти моніторингу повинні надавати детальні показники на рівні обладнання, такі як стан диска, температура процесора, помилки пам'яті та використання енергії. Така видимість допомагає командам перевіряти стабільність продуктивності та виявляти ранні ознаки деградації обладнання, забезпечуючи надійну роботу відновленого обладнання разом з новішими системами.

Метрики продуктивності складають основу ефективного моніторингу. Використання процесора допомагає виявляти перевантажені сервери, а споживання пам'яті запобігає уповільненням, спричиненим виснаженням ресурсів. Моніторинг дискового простору гарантує, що бази даних і програми продовжують працювати без обмежень щодо сховища. Сповіщення в режимі реального часу дозволяють командам швидко реагувати, коли перевищено порогові значення продуктивності.

Бюджетні міркування виходять за рамки початкових витрат. Рішення з відкритим кодом пропонують гнучкість та економію коштів, але можуть вимагати технічної експертизи для управління. Корпоративні платформи забезпечують автоматизацію, штучний інтелект та спеціалізовану підтримку, що зменшує операційне навантаження.

Можливості безпеки та інтеграції однаково важливі. Інструменти повинні підтримувати шифрування, контроль доступу на основі ролей та інтеграцію з платформами ITSM, такими як ServiceNow або Jira. Тестування інструментів за допомогою безкоштовних пробних версій допомагає забезпечити сумісність та зручність використання перед повним розгортанням.

Надійний моніторинг серверів став важливим для організацій, які покладаються на цифрову інфраструктуру для обслуговування клієнтів та виконання критично важливих операцій. Оскільки середовища стають більш

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		24

розподіленими та динамічними, видимість у режимі реального часу допомагає командам виявляти проблеми з продуктивністю на ранній стадії та підтримувати стабільний час безвідмовної роботи. Правильна платформа моніторингу дозволяє командам розуміти, як системи поведуться під навантаженням, виявляти приховані вузькі місця та швидко реагувати, перш ніж незначні проблеми переростуть у перебої в обслуговуванні.

Найефективніша стратегія моніторингу зосереджена на точності, своєчасних сповіщеннях та чіткій діагностичній аналітиці. Коли команди мають доступ до надійних даних про продуктивність, вони можуть швидше усувати несправності, оптимізувати ресурси та впевнено планувати зміни в інфраструктурі. Інвестування в потужне рішення для моніторингу знижує операційні ризики, підвищує надійність послуг та допомагає забезпечити стабільність систем, оскільки бізнес-вимоги та технічні вимоги продовжують зростати.

2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

Python – це об'єктно-орієнтована мова програмування високого рівня загального призначення з відкритим кодом. Це визначення може бути важким для новачків, тому розглянемо кожну характеристику окремо, щоб зрозуміти, що вона означає:

- Відкритий вихідний код: це безкоштовно та доступно для подальших покращень, таких як додавання корисних функцій або виправлення помилок.
- Об'єктно-орієнтована: заснована не на функціях, але в об'єктах з певними атрибутами й методами.
- Високий рівень: зручний для людини, а не для комп'ютера.
- Загальне призначення: можна використовувати для створення будь-яких програм.

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		25

Ця мова використовується в будь-якому програмному забезпеченні, про яке ви тільки можете подумати. Ви можете використовувати його для створення веб-сайтів, штучного інтелекту, серверів, програмного забезпечення для бізнесу та багато іншого. Також застосовується в науці про дані, аналізі даних, машинному навчанні, інженерії даних, веб-розробці, розробці програмного забезпечення та інших галузях.

Переваги та недоліки Python

Переваги:

– Її легко читати, вчити та писати. Це мова програмування високого рівня з англійським синтаксисом. Це полегшує читання та розуміння коду. Її дійсно легко зрозуміти і вивчити, тому багато людей рекомендують Python новачкам. Вам потрібно менше рядків коду для виконання того ж завдання в порівнянні з іншими основними мовами, такими як C/C++ та Java.

– Підвищує продуктивність. Це дуже продуктивна мова. Завдяки її простоті розробники можуть зосередитися на розв'язанні проблеми. Їм не потрібно витрачати багато часу на розуміння синтаксису або поведінку мови програмування. Ви пишете менше коду та виконуєте більше завдань.

– Інтерпретована мова. Python мова, що інтерпретується, а це означає, що вона безпосередньо виконує код по рядку. Якщо сталася помилка, вона зупиняє подальше виконання та повідомляє про її виникнення. Вона показує лише одну помилку, навіть якщо у програмі їх кілька. Це спрощує налагодження.

– Динамічно типізована. Python не визначає тип змінної, доки ми не запустимо код. Вона автоматично надає тип даних, коли відбувається процес виконання. Фахівець може не турбуватися про оголошення змінних та типи даних.

– Безкоштовна та з відкритим вихідним кодом. Ця мова постачається під схваленою OSI ліцензією з відкритим вихідним кодом. Це робить його безкоштовним для використання та розповсюдження. Ви можете завантажити вихідний код, змінити його та навіть розповсюджувати свою версію. Це корисно для організацій, які хочуть використати свою версію для розробки.

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		26

– Підтримка великих бібліотек. Стандартна бібліотека Python є величезною, ви можете знайти майже всі функції, необхідні для вашого завдання. Таким чином ви не залежите від зовнішніх бібліотек.

– Портативність. У багатьох мовах, таких як C/C++, потрібно змінити свій код, щоб запустити програму на різних платформах. З Python все інакше. Ви тільки пишете один раз і запускаєте її будь-де.

Недоліки:

– Низька швидкість. Вище ми обговорювали, що це інтерпретована мова з динамічною типізацією. Порядкове виконання коду часто призводить до повільного виконання. Динамічна природа Python також є причиною її низької швидкості, оскільки їй доводиться виконувати додаткову роботу при виконанні коду. Тому вона не підходить для цілей, де швидкість важливий аспект проєкту.

– Неефективна для пам'яті. Ця мова програмування використовує великий обсяг пам'яті, це може бути недоліком при створенні програм, коли віддають перевагу оптимізації пам'яті.

– Слабка у мобільних обчисленнях. Python зазвичай використовується у серверному програмуванні. Ми не бачимо – її на стороні клієнта або в мобільних програмах з таких причин: вона не заощаджує пам'ять і має повільну обчислювальну потужність у порівнянні з іншими мовами.

– Доступ до бази даних. Програмувати на цій мові легко, але коли ми взаємодіємо з базою даних, її не вистачає. Рівень доступу до бази даних у Python примітивний та недостатньо розвинений у порівнянні з іншими популярними технологіями.

– Помилки виконання. Це мова з динамічною типізацією, тому тип даних змінної може змінюватись у будь-який час. Змінна, що містить ціле число, у майбутньому може містити рядок, що може призвести до помилок виконання.

Застосування Python:

– Для аналізу даних. Дані стали цінним активом у будь-якій сучасній галузі, і більшість компаній зацікавлені у збиранні, обробці та аналізі релевантних даних, щоб витягти з них цінну інформацію для бізнесу. І тут Python виходить за межі будь-якої конкуренції. Python особливо цінна тим, що крім

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		27

великої стандартної бібліотеки надає величезний набір додаткових модулів, розроблених спеціально для аналітичних цілей. Найвідоміші бібліотеки Python для аналізу даних – це pandas і NumPy. Ці інструменти дозволяють робити з вашими даними майже все, наприклад, очищати і аналізувати їх, вивчати статистику або візуалізувати приховані тенденції у ваших даних.

– Для візуалізації даних. Візуалізація даних – це окрема частина аналізу даних, яка допомагає нам подавати інформацію, необроблену чи очищену, у більш змістовній формі. Тут Python знову входить у гру, пропонуючи широкий спектр інструментів візуалізації даних. Найпопулярніші з них – matplotlib і заснований на ній seaborn. Використовуючи їх, ми можемо створювати буквально всі види візуалізації: від найпростіших до складніших.

– Для машинного навчання. Машинне навчання (ML) є основою більшості завдань науки даних. Він є областю штучного інтелекту, пов'язаною з використанням алгоритмів, що дозволяють машинам вивчати закономірності та тенденції на основі історичних даних, щоб робити прогнози на основі невідомих даних. – Використовуючи методи ML, ми можемо створювати моделі, які можуть точно передбачити швидкість відтоку клієнтів компанії, оцінити ризик виникнення у людини певного захворювання, визначити оптимальне розташування автомобілів таксі й т.д. За допомогою Python ми можемо побудувати модель ML, використовуючи лише три рядки коду.

– Для розробки програмного забезпечення. Крім свого багатостороннього застосування в галузях науки про дані, Python використовується на кожному етапі розробки програмного забезпечення, включаючи контроль складання, автоматичну безперервну компіляцію, прототипування, відстеження помилок, тестування та обслуговування програмного забезпечення. За допомогою цієї мови можемо створювати аудіо- або відеопрограми на основі методів штучного інтелекту, машинного навчання, API (інтерфейсів прикладного програмування), GUI (графічних інтерфейсів) або будь-якого іншого типу програмного забезпечення.

– Для веброзробки. У той час як для створення візуальної частини вебсайту ми переважно будемо використовувати такі мови, як HTML, CSS та

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		28

JavaScript, для його невидимої частини ми часто вибираємо Python. Серед масштабних вебсайтів та програм, створених за допомогою цієї мови, варто згадати Google, Facebook, Instagram, YouTube, Dropbox та Reddit.

– Для автоматизації задач/скриптингу. Це відмінний інструмент для написання програм для автоматизації різних завдань, що повторюються. Цей процес називається скриптингом. Зокрема, можна робити скрипти для роботи з файлами та папками. Наприклад, можна створювати, перейменовувати, перетворювати, розділяти, об'єднувати або видаляти файли, перевіряти їх наявність помилок. Ви також можете використовувати автоматизацію Python для пошуку та завантаження інформації з Інтернету, заповнення та надсилання онлайн-форм та надсилання регулярних повідомлень або електронних листів.

Яким фахівцям потрібно володіти Python:

- Фахівець з даних.
- Аналітик даних.
- Інженер даних.
- Інженер з машинного навчання.
- Журналіст даних.
- Архітектор даних.
- Повний стек веб-розробника.
- Backend-розробник.
- DevOps-інженер.
- Інженер-програміст.

Можемо зробити висновок, що Python ще довго буде популярною мовою, хоч і має низку недоліків. Цю мову використовують для створення вебсайтів, штучного інтелекту, серверів, програмного забезпечення для бізнесу, аналізу даних, машинного навчання, інженерії даних та для багатьох інших областей. Це перспективна і затребувана навичка, яка необхідна у всіх галузях.

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускню кваліфікаційну роботу за першим (бакалаврським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи інтелектуального моніторингу та оптимізації ІТ-систем ЦОД.

В процесі розробки випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти необхідно виконати наступний обсяг роботи:

а) провести аналіз існуючих систем-аналогів для виявлення їх позитивних і негативних якостей. Результати аналізу врахувати в подальших розробках;

б) вибрати та обґрунтувати методику побудови системи контролю роботи технологічного обладнання на виробництві в автоматизованому режимі. Розробити функціональну та структурну схеми системи;

в) розробити програмне забезпечення системи, що дозволить реалізувати поставлену технічним завданням задачу. Побудувати блок-схеми алгоритмів програми та підпрограми;

г) організувати інтерфейс користувача з метою формування та виводу на екран ЕОМ повідомлень про некоректні дії користувача та нестандартні ситуації в роботі технологічного обладнання;

д) розробити рекомендації по організаційних та методичних заходах, які забезпечать впровадження системи в промислову експлуатацію та її подальшу успішну експлуатацію;

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

ж) розробити заходи по охороні праці при впровадженні та експлуатації системи, а також розробити заходи з цивільного захисту;

з) сформулювати висновки про виконаний обсяг робіт та одержані результати.

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		30

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

Оскільки бізнес продовжує впроваджувати цифрову трансформацію, центри обробки даних стали основою сучасної ІТ-інфраструктури. Від хмарних обчислень до аналітики великих даних – все залежить від ефективної та масштабованої роботи центрів обробки даних. У 2026 році управління центрами обробки даних зазнає значних змін, зумовлених штучним інтелектом (ШІ), автоматизацією та сталим розвитком.

Зараз організації зосереджуються на розумніших, швидших та екологічніших центрах обробки даних, щоб задовольнити зростаючий попит. У цьому розділі досліджується, як розвивається майбутнє управління центрами обробки даних та які тенденції формують галузь.

Роль штучного інтелекту в управлінні центрами обробки даних

Штучний інтелект революціонізує роботу центрів обробки даних. Системи на базі штучного інтелекту можуть контролювати, аналізувати та оптимізувати операції в режимі реального часу.

Основні переваги:

- Прогнозне технічне обслуговування для уникнення простоїв.
- Автоматизований розподіл ресурсів.
- Інтелектуальне балансування робочого навантаження.
- Моніторинг продуктивності в режимі реального часу.

Штучний інтелект допомагає зменшити втручання людини, одночасно підвищуючи ефективність та надійність.

Зростання хмарних центрів обробки даних

Хмарні обчислення продовжують домінувати в ІТ-ландшафті у 2026 році. Бізнес переходить від традиційних локальних центрів обробки даних до хмарних

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		31

рішень.

Переваги:

- Масштабованість на вимогу.
- Економічна ефективність.
- Віддалений доступ.
- Швидше розгортання.

Гібридні та мультимарні середовища стають новим стандартом для підприємств.

Автоматизація та інтелектуальні операції

Автоматизація є ключовою тенденцією в сучасному управлінні центрами обробки даних. Від налаштування серверів до управління мережею, інструменти автоматизації спрощують складні процеси.

Ключові напрямки автоматизації:

- Розгортання сервера.
- Конфігурація мережі.
- Моніторинг безпеки.
- Резервне копіювання та відновлення.

Автоматизація зменшує кількість помилок ручного керування та підвищує швидкість роботи.

Акцент на енергоефективності та сталому розвитку

Зі зростанням екологічної стурбованості, зелені центри обробки даних набувають все більшого значення.

Ключові інновації:

- Енергоефективні системи охолодження.
- Використання відновлюваних джерел енергії.
- Розумне управління живленням.
- Зменшення вуглецевого сліду.

Компанії інвестують у стійкі рішення для зменшення операційних витрат та впливу на навколишнє середовище.

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		32

Посилена безпека та кіберстійкість

Зі зростанням кіберзагроз безпека центрів обробки даних стає все більш критичною.

Тенденції безпеки:

- Виявлення загроз на основі штучного інтелекту.
- Моделі безпеки з нульовою довірою.
- Розширені методи шифрування.
- Моніторинг загроз у режимі реального часу.

Майбутні центри обробки даних будуть більш стійкими до кібератак та витоків даних.

Периферійні обчислення та децентралізація

Периферійні обчислення змінюють способи обробки та зберігання даних.

Переваги:

- Зменшена затримка.
- Швидша обробка даних.
- Покращений користувацький досвід.
- Краща підтримка пристроїв Інтернету речей.

Замість того, щоб покладатися виключно на централізовані центри обробки даних, компанії розгортають периферійні центри обробки даних ближче до користувачів.

Можливості працевлаштування в центрах обробки даних у 2026 році

З розвитком технологій з'являються нові посади в управлінні центрами обробки даних:

- Інженер центру обробки даних.
- Хмарний архітектор.
- Спеціаліст з операцій зі штучного інтелекту.
- Аналітик мережевої безпеки.
- Інженер з автоматизації інфраструктури.

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		33

Фахівці з навичками штучного інтелекту, хмарних обчислень та кібербезпеки будуть дуже затребувані.

Проблеми управління центрами обробки даних

Незважаючи на досягнутий прогрес, організації стикаються з кількома проблемами:

- Високе енергоспоживання.
- Ризики безпеки даних.
- Управління складною інфраструктурою.
- Дефіцит кваліфікованої робочої сили.

Вирішення цих викликів має вирішальне значення для майбутнього зростання.

Майбутні тенденції, на які варто звернути увагу:

- Автономні центри обробки даних на базі штучного інтелекту.
- Інтеграція квантових обчислень.
- Програмно-визначена інфраструктура.
- Технологія цифрових двійників.
- Центри обробки даних з підтримкою 5G.

Ці інновації переосмислять те, як функціонують центри обробки даних у найближчі роки.

Майбутнє управління центрами обробки даних у 2026 році залежить від штучного інтелекту, автоматизації та сталого розвитку. Оскільки бізнес продовжує покладатися на цифрову інфраструктуру, центри обробки даних повинні розвиватися, щоб стати більш ефективними, безпечними та масштабованими.

Організації, які впроваджують сучасні технології на ранній стадії, отримають конкурентну перевагу, тоді як фахівці, які підвищують свою кваліфікацію, знайдуть захопливі кар'єрні можливості в цій зростаючій галузі.

3.2 Розробка структурної схеми

У даному розділі досліджуються операційні та технічні тригери, які вказують на те, як команди з експлуатації центрів обробки даних переросли свої основні інструменти та платформи моніторингу, а також окреслюється основа для розвитку в напрямку спостереження, аналітики та інтелектуального оповіщення корпоративного рівня. Сучасний програмний стек для центрів обробки даних часто будується на платформах, що перетинаються:

- Системи управління будівлями (BMS).
- Системи моніторингу електроенергії (EPMS).
- Управління інфраструктурою центру обробки даних (DCIM).

Ці системи часто використовують спільні пристрої, панелі інструментів та телеметрію. Оператори часто бачать однакові сигнали тривоги, електричні однолінійні показники, показники ДБЖ, дані генератора, показники температури та тенденції живлення, що реплікуються на кількох інструментах.

Хоча кожна платформа виконує важливу операційну функцію, вони здебільшого функціонують як системи обліку, а не як системи розуміння. І це створює проблему. Якщо кілька платформ показують однакові дані моніторингу інфраструктури, чи отримують команди насправді глибше операційне розуміння, чи просто дублюють видимість?

Багато об'єктів залишаються в пастці між реактивним моніторингом та справжньою оперативною аналітикою. Обмеження традиційних інструментів моніторингу зазвичай включають:

- Статичні порогові сигналізації.
- Аналіз ізольованих трендів.
- Обмежена міжсистемна кореляція.
- Високий рівень шуму тривоги.
- Відсутність прогнозової аналітики.
- Мінімальна контекстуальна обізнаність.

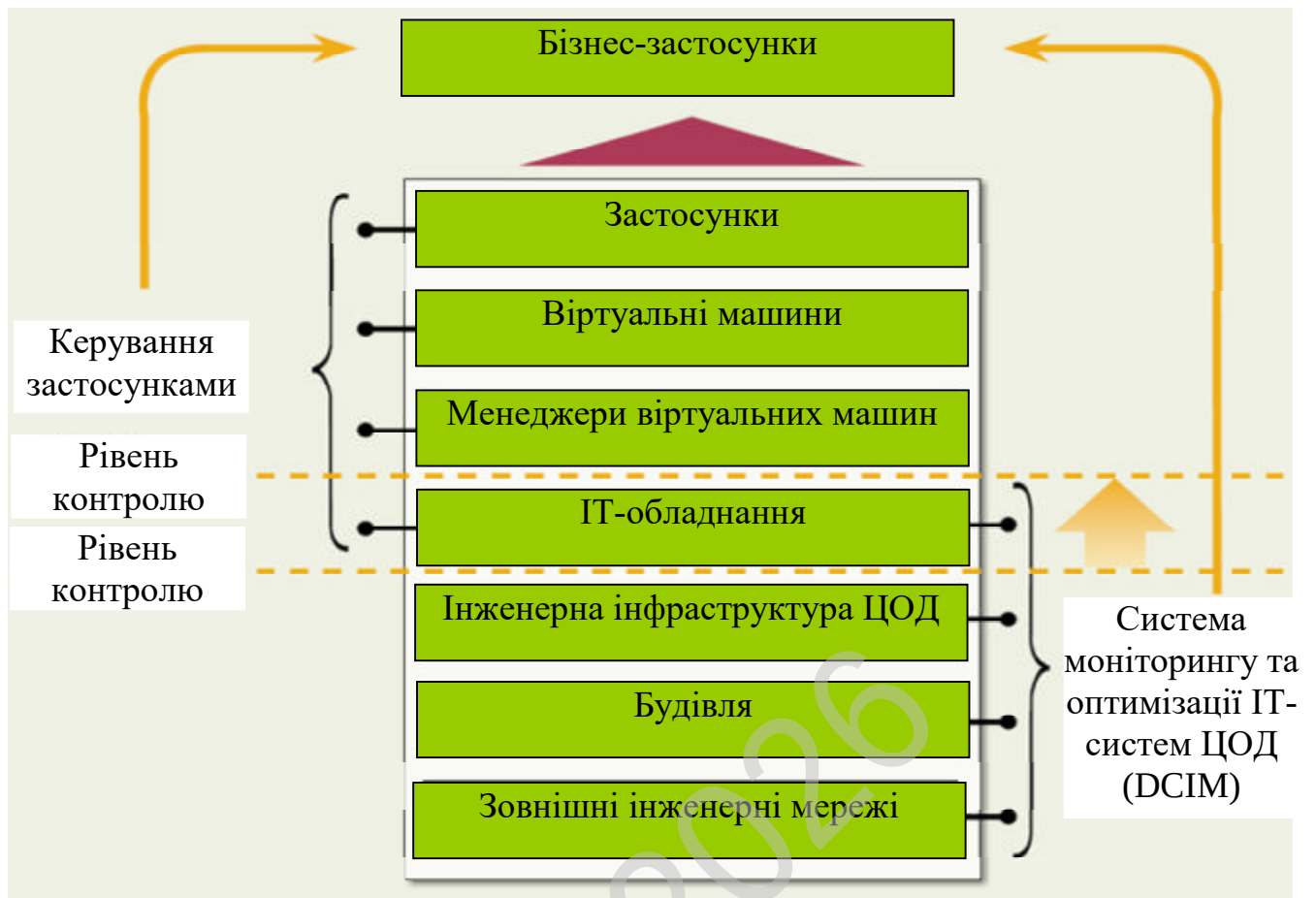


Рисунок 3.1 – Структурна схема системи

На практиці це означає, що команди витрачають більше часу на реагування на сповіщення, ніж на проактивну оптимізацію продуктивності інфраструктури.

Проблема рідко полягає в нестачі даних. Проблема полягає в масштабованості моніторингу інфраструктури – здатності контекстуалізувати мільйони точок телеметрії в практичну оперативну аналітику.

Корисною основою для оцінки вимог до моніторингу підприємства є розгляд рівня зрілості моніторингу на п'яти операційних рівнях, як відображено у таблиці 3.1.

Більшість середовищ все ще працюють десь на рівні 2,5 – оточені спостережуваністю та показниками продуктивності, але без змістовної операційної інтерпретації.

Таблиця 3.1 – П'ять рівнів зрілості моніторингу

Рівень	Категорія	Можливості	Операційний результат
1	Реактивний	Сигнали тривоги бінарних пристроїв	Реакція на виправлення помилок
2	На основі порогу	Базовий моніторинг верхнього/нижнього рівня та тренди	Покращена усвідомленість, але все ще реагування
3	Контекстуальна усвідомленість	Аналіз міжсистемних зв'язків	Краще прийняття операційних рішень
4	Прогностичний/Профілактичний	Прогнозування ризиків та виявлення аномалій	Проактивне обслуговування та запобігання збоям
5	Призначальний	Інтелектуальні рекомендації щодо оптимізації	Автоматизоване керування ефективністю та пропускнуою здатністю

П'ять ключових операційних попереджувальних ознак свідчать про те, що організації переросли традиційне програмне забезпечення для моніторингу центрів обробки даних.

1. Зростає втома від сповіщень

Якщо команди отримують велику кількість сповіщень без чіткої пріоритезації або контексту першопричин, платформа моніторингу може більше не підтримувати ефективне оповіщення та управління інцидентами.

2. Моніторинг здійснюється ізольовано

Коли платформи BMS, EPMS та DCIM не можуть співвіднести операційні зв'язки між системами охолодження, живлення та ІТ, організації втрачають здатність цілісно розуміти взаємодію інфраструктури.

3. Оптимізація охолодження залишається ручною

Команди об'єктів часто покладаються на ручне налаштування потоку повітря, поведінки CRAC або конфігурацій охолодженої води, оскільки платформам моніторингу бракує чіткої аналітики.

4. Рішення щодо потужності несуть високий ризик

Без прогнозного аналізу розширення інфраструктури та розгортання робочих навантажень стають надмірно консервативними або операційно ризикованими.

5. Відсутність контексту для показників сталого розвитку

Показники PUE, WUE та ефективності охолодження можуть бути доступними, але без аналітики на системному рівні командам важко визначити дієві можливості для оптимізації.

Спроектоване у даній роботі програмне забезпечення було розроблено спеціально для подолання цього операційного розриву.

Замість того, щоб замінити існуючі інвестиції в BMS, EPMS або DCIM, розроблене у даній роботі програмне забезпечення впроваджує систему розуміння в усьому середовищі центру обробки даних.

Платформа виходить за рамки традиційних показників спостереження та продуктивності, поєднуючи:

- 3D-візуалізація цифрового двійника.
- Теплова аналітика на основі штучного інтелекту.
- Прогнозне виявлення аномалій.
- Інтелектуальна оптимізація охолодження.
- Аналітика розгортання потужностей.
- Аналітика чилерної системи.

На відміну від багатьох нових платформ, які намагаються впровадити можливості штучного інтелекту в застарілі системи, машинне навчання та розширена аналітика є основою даної системи.

Контекстуальна обізнаність

Розроблене у даній роботі програмне забезпечення співвідносить операційні зв'язки між системами, зокрема:

- Кореляція PUE з умовами навколишнього середовища.
- Моніторинг відповідності ASHRAE.
- Аналіз режиму охолодження.
- Картування теплових зв'язків.

Прогнозний та превентивний аналіз

Платформа виявляє аномальну поведінку системи, перш ніж вона переросте в операційний ризик, за допомогою:

- Розширене виявлення аномалій.
- Аналіз зон впливу CRAN.
- Прогнозне моделювання теплової поведінки.

Прескриптивна оптимізація

Найголовніше, що розроблене у даній роботі програмне забезпечення перетворює моніторинг на практичний інтелект завдяки:

- Рекомендації щодо оптимізації охолодження.
- Керівництво з розгортання потужностей.
- Підтримка операційних рішень з урахуванням ризиків.

Майбутні вимоги до моніторингу підприємств

Оскільки центри обробки даних продовжують розвиватися для підтримки робочих навантажень штучного інтелекту, підвищення щільності стійок та забезпечення сталого розвитку, вимоги до моніторингу підприємств дедалі більше вимагатимуть більше, ніж просто видимості.

Операторам потрібні платформи, здатні:

- Кореляція поведінки інфраструктури в різних доменах.

- Зменшення операційної складності.
- Підтримка інтелектуального оповіщення та управління інцидентами.
- Надання прогностичного оперативного керівництва.
- Оптимізація ефективності без збільшення ризику.

Майбутнє належить системам розуміння, а не просто системам запису.

Розроблене у даній роботі програмне забезпечення не замінює цінну прозорість, яку вже забезпечують платформи BMS, EPMS або DCIM. Натомість, вона посилює та доповнює їх.

Перетворюючи операційну телеметрію на контекстну аналітику щодо потужності, ефективності та ризиків, розроблене у даній роботі програмне забезпечення дозволяє командам центрів обробки даних приймати швидші, розумніші та впевненіші рішення – без копій та заміни проєктів чи додаткового технічного боргу. Дані завжди існували. Конкурентна перевага полягає в їх кращому розумінні та застосуванні на практиці.

3.3 Розробка функціональної схеми

Функціональна схема розробленої системи, показана на рисунку 3.2, складається із наступних блоків:

- центрального блоку;
- агентів спостереження.

Центральний блок складається з наступних блоків, які взаємодіють один з одним згідно структурної схеми:

- Планувальник, який визначає які зв'язки потрібно коригувати, або які вузли необхідно внести.
- База даних.
- Блок інтерфейсу ручного вводу, призначений для ручного вводу ймовірного трафіку у тому, або іншому вузлі мережі центрів обробки даних на основі технології DCIM.

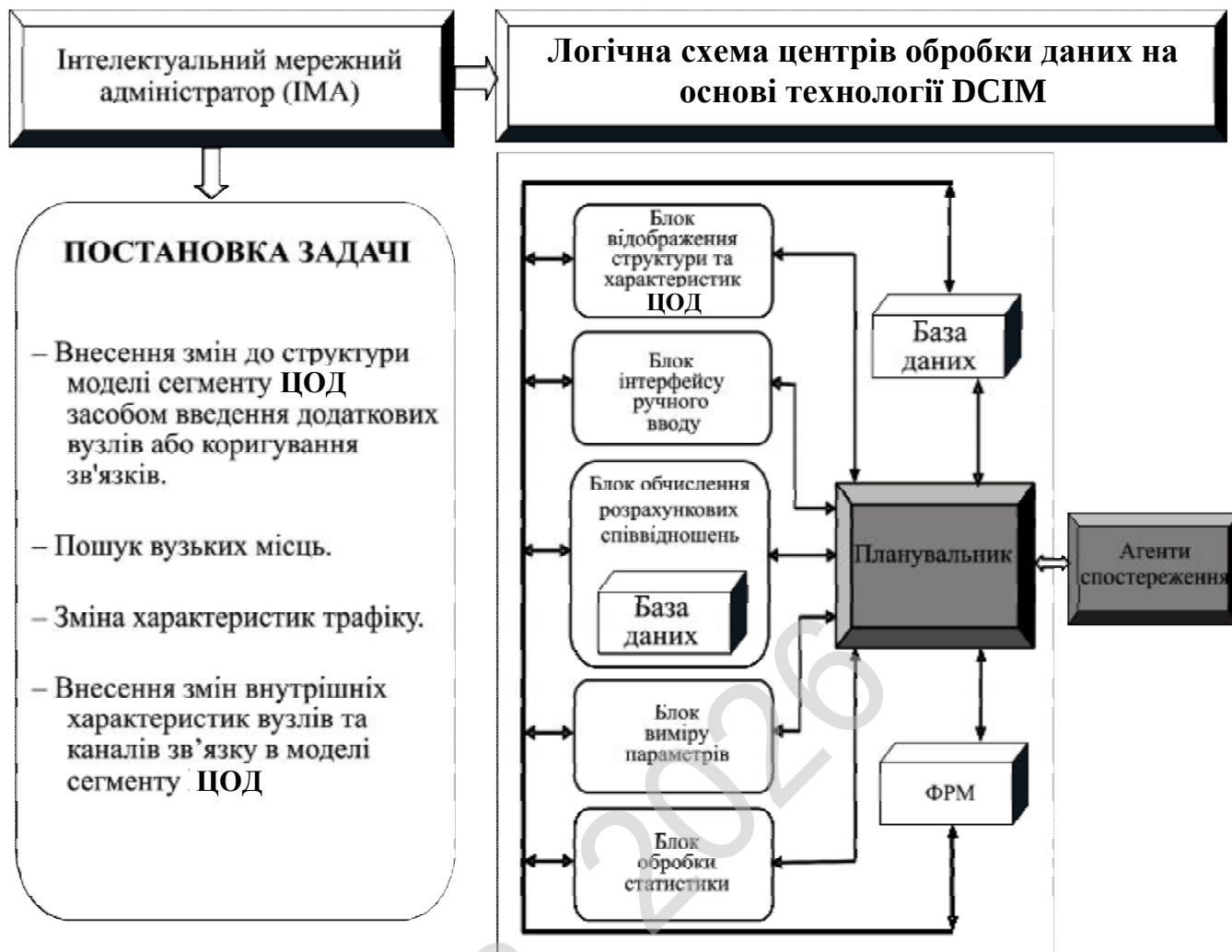


Рисунок 3.2 – Функціональна схема системи

– Блок відображення структури та характеристик мережі центрів обробки даних на основі технології DCIM, для наглядного бачення усієї мережі центрів обробки даних на основі технології DCIM, він може бути виконаний як на логічному так і на фізичному рівні представлення мережі центрів обробки даних на основі технології DCIM.

– Блок обчислення розрахункових значень, призначений для визначення необхідності зміни топології мережі центрів обробки даних на основі технології DCIM.

– Блок виміру параметрів, призначений для виміру параметрів мережі центрів обробки даних на основі технології DCIM.

– Блок обробки статистики, передачі трафіку у мережі центрів обробки даних на основі технології DCIM.

– Формули розрахунку мережі центрів обробки даних на основі технології DCIM (ФРМ), які дозволяють на основі використання розробленої математичної моделі мережі центрів обробки даних на основі технології DCIM, з використанням методів та формул теорії масового обслуговування та теорії статистики, розраховувати навантаження трафіку на кожний вузол мережі центрів обробки даних на основі технології DCIM.

БД використовується для зберігання таблиць, що містять довідкову інформацію й дані, отримані в ході вимірів. Розроблена специфікація моделі й структура бази даних, підтверджують реалізуємість математичної моделі, вбудованої в ІМА, для мережі центрів обробки даних на основі технології DCIM реальної розмірності

Взаємодія блоків інтелектуального адміністратора мережі центрів обробки даних на основі технології DCIM складається з:

– Бази даних (БД) – блоку, що містить постійно оновлювані дані про мережу.

– Блоку обчислення розрахункових співвідношень (БРС), призначеного для обчислення характеристик мережі центрів обробки даних на основі технології DCIM.

– Блоку знань (БЗ), що містить формули й характеристики законів розподілів, які буде використовувати для роботи БРС.

– Блоку відображення структури й характеристик, призначеного для організації взаємодії мережного адміністратора з адміністратором (А) системи.

Розглянувши усі блоки функціональної схеми перейдемо до розгляду діаграми взаємодії процесів, які відбуваються у системі.

3.4 Розробка діаграми процесів

Діаграма процесів розробленої системи зображена на рисунку 3.3. При детальному її розгляді можна побачити як саме проходить взаємодія у розробленій системі.



Рисунок 3.3 – Діаграма взаємодії процесів

Використовується модель проектування, графічне представлення «потоків» даних в інформаційній системі.

Діаграма взаємодії процесів використовується для візуалізації процесів обробки даних (структурне проектування).

Для розробника вважається звичним спочатку креслити діаграму взаємодії процесів даних рівня контексту, завдяки чому буде показано взаємодію системи. Ця діаграма в подальшому підлягає уточненню шляхом деталізації процесів та потоків даних з метою показати систему що розробляється.

Діаграми потоків даних містять чотири типи елементів:

– Процеси які являють собою трансформацію даних в рамках описуваної системи.

– Сховища даних (репозиторії).

– Зовнішні по відношенню до системи сутності.

– Потоки даних між елементами трьох попередніх типів.

Таким чином, розглянувши опис системи, структурну, функціональну схеми системи, та діаграму взаємодії процесів перейдемо до опису блок-схем основної програми, та підпрограм, які використовуються, для реалізації системи.

К6ПЗ_2026

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		44

4 РЕАЛІЗАЦІЯ ПРОЕКТУ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ПРАВИЛЬНІСТЬ ПРОЕКТНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

Під час роботи над кваліфікаційною бакалаврською роботою було створено блок-схеми. Перед їх розглядом необхідно провести роз'яснення який саме тип блок-схем використовується.

Блок-схема це представлення задачі для її аналізу або розв'язування за допомогою спеціальних символів (геометричних образів), які позначають такі елементи, як операції, потік, дані тощо. Блок вхідних та вихідних даних прийнято позначати паралелограмом, блок обчислень (обробки) даних – прямокутником, блок прийняття рішень – ромбом, еліпсом – початок та кінець алгоритму.

У інформаційних технологіях функціональна схема складається з функціональних блоків, які являють собою конструктивно відособлені частини (елементи або пристрої) автоматичних систем, які виконують певні функції. Функціональні блоки на схемі позначають прямокутниками, всередині яких надписують їх найменування відповідно до функцій, що виконуються. Зв'язки між функціональними блоками (внутрішні впливи) позначаються лініями зі стрілками, які вказують напрям впливів.

Функціональні схеми можуть виконуватися в укрупненому і розгорненому вигляді. У першому випадку на схемі зображають найважливіші блоки системи і зв'язки між ними.

У другому варіанті схема відображається більш детально, що полегшує її читання та ілюструє принцип роботи.

Основні елементи схем алгоритму це термінатор, процес, рішення, зумовлений процес (підпрограма), дані та з'єднувач.

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		45

Термінатор це елемент відображає вхід із зовнішнього середовища або вихід з неї (найчастіше застосування – початок і кінець програми). Всередині фігури записується відповідна дія.

Процес це виконання однієї або кількох операцій, обробка даних будь-якого виду (зміна значення даних, форми подання, розташування). Всередині фігури записують безпосередньо самі операції.

Рішення це показує рішення або функцію перемикального типу з одним входом і двома або більше альтернативними виходами, з яких тільки один може бути обраний після обчислення умов, визначених всередині цього елемента. Вхід в елемент позначається лінією, що входить зазвичай у верхню вершину елемента. Якщо виходів два чи три то зазвичай кожен вихід позначається лінією, що виходить з решти вершин (бічних і нижній). Якщо виходів більше трьох, то їх слід показувати однією лінією, що виходить з вершини (частіше нижній) елемента, яка потім розгалужується. Відповідні результати обчислень можуть записуватися поруч з лініями, що відображають ці шляхи.

Зумовлений процес (підпрограма) це символ відображає виконання процесу, що складається з однієї або кількох операцій, що визначені в іншому місці програми (у підпрограмі, модулі). Всередині символу записується назва процесу і передані в нього дані.

Дані це перетворення у форму, придатну для обробки (введення) або відображення результатів обробки (виведення). Цей символ не визначає носія даних (для вказівки типу носія даних використовуються специфічні символи).

З'єднувач це символ відображає вихід в частину схеми і вхід з іншої частини цієї схеми. Використовується для обриву лінії та продовження її в іншому місці (приклад: поділ блок-схеми, що не поміщається на листі). Відповідні сполучні символи повинні мати одне (при тому унікальне) позначення.

Блок-схеми є першоджерелами стратегії розвитку ПЗ. Тому від точності і детальної блок-схеми залежить результат всієї програми. При виборі початкової точки відліку при побудові схем було враховано, що виходячи з вибору мови

програмування і інших технічних засобів, програма буде об'єктно-орієнтована що вимагає оптимізації програми високого рівня, також те, що при розробці програми слід надати особливу увагу модулю системи моніторингу та оптимізації ІТ-систем ЦОД.

На рисунку 4.1 зображена основна блок-схема програми, на рисунку 4.2 зображено роботу підпрограми.

З яких видно що робота основної програми складається з початкових етапів ініціалізації ПЗ, перевірки наявності ресурсів системи, блоку початку основного циклу з чеканням запиту від користувача в якому відбувається виклик підпрограми та останньої стадії – перевірка поточного стану з завершенням роботи розробленого ПЗ. При роботі підпрограми виконується основний функціонал системи з циклічними послідовностями, перевіркою поточного стану та поверненням в основну програму прапорів стану виконання.

Було використано підходи з використанням UML, це уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого програмування. Є невід'ємною частиною уніфікованого процесу розробки програмного забезпечення. UML є мовою широкого профілю, це відкритий стандарт, що використовує графічні позначення для створення абстрактної моделі системи, називаної UML-моделлю.

UML був створений для визначення, візуалізації, проектування й документування в основному програмних систем. UML не є мовою програмування, але в засобах виконання UML-моделей як інтерпретованого коду можлива кодогенерація.

UML може бути застосовано на всіх етапах життєвого циклу аналізу бізнес-систем і розробки прикладних програм. Різні види діаграм які підтримуються UML, і найбагатший набір можливостей представлення певних аспектів системи робить UML універсальним засобом опису як програмних, так і ділових систем.

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		47

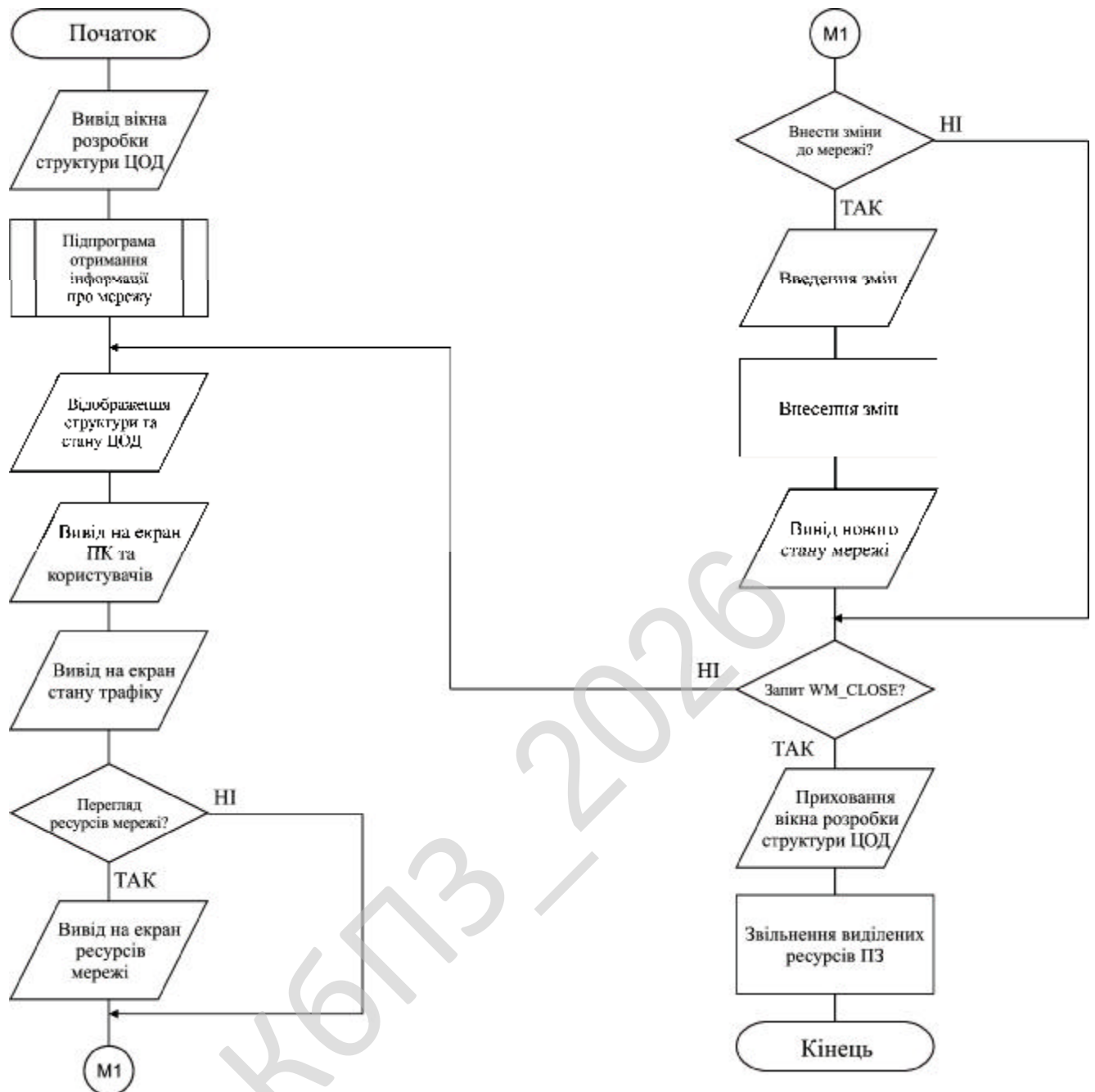


Рисунок 4.1 – Блок-схема основної програми

Діаграми дають можливість представити систему (як ділову, так і програмну) у такому вигляді, щоб її можна було легко перевести в програмний код. Основною причиною використання мови UML є спілкування розробників між собою.

Крім того, UML спеціально створювалася для оптимізації процесу розробки програмних систем, що дозволяє збільшити ефективність їх реалізації у кілька разів і помітно поліпшити якість кінцевого продукту.

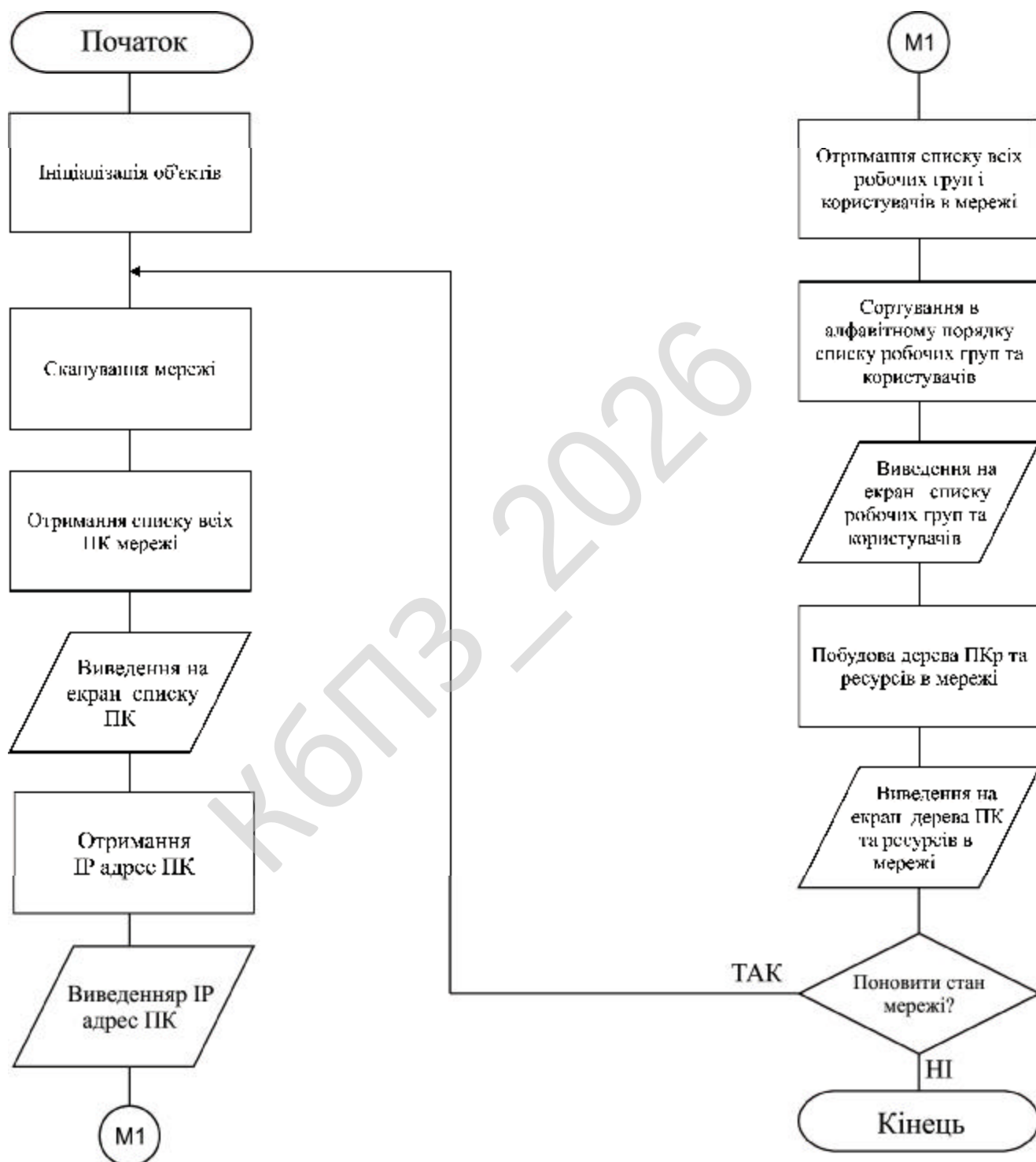


Рисунок 4.2 – Блок-схема роботи підпрограми

UML прекрасно зарекомендувала себе в багатьох успішних програмних проектах. Засоби автоматичної генерації кодів дозволяють перетворювати моделі мовою UML у вихідний код об'єктно-орієнтованих мов програмування, що ще більш прискорює процес розробки. Практично усі CASE-засоби (програми автоматизації процесу аналізу і проектування) мають підтримку UML. Моделі розроблені в UML, дозволяють значно спростити процес кодування і направити зусилля програмістів безпосередньо на реалізацію системи.

Діаграми підвищують супроводжуваність проекту і полегшують розробку документації.

UML необхідний:

- Керівникам проектів, які керують розподілом завдань і контролем за проектом.
- Проектувальникам інформаційних систем які розробляють технічні завдання для програмістів.
- Бізнес-аналітикам, які досліджують реальну систему і здійснюють інжиніринг і реінжиніринг бізнесу компанії.
- Програмістам які реалізують модулі інформаційної системи.

При модифікації системи об'єктний підхід дозволяє легко включати в систему нові об'єкти і виключати застарілі без істотної зміни її життєздатності. Використання побудованої моделі при модифікаціях системи дає можливість усунути небажані наслідки змін, оскільки вони не ламають структури системи, а тільки змінюють поведінку об'єктів.

Також при розробці бакалаврської дипломної роботи було використано наступні підходи UML: діаграма діяльності (діаграми поведінки типу); діаграма прецедентів (діаграми поведінки типу); Діаграма класів; Діаграма компонент; Діаграма об'єктів; Діаграма розгортання.

Діаграма діяльності. Це візуальне представлення графу діяльностей. Граф діяльностей є різновидом графу станів скінченного автомату, вершинами якого є певні дії, а переходи відбуваються по завершенню дій. Дія є фундаментальною

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		50

одиницею визначення поведінки в специфікації. Дія отримує множину вхідних сигналів, та перетворює їх на множину вихідних сигналів.

Одна із цих множин, або обидві водночас, можуть бути порожніми. Виконання дії відповідає виконанню окремої дії. Подібно до цього, виконання діяльності є виконанням окремої діяльності, буквально, включно із виконанням тих дій, що містяться в діяльності. Кожна дія в діяльності може виконуватись один, два, або більше разів під час одного виконання діяльності. Щонайменше, дії мають отримувати дані, перетворювати їх та тестувати, деякі дії можуть вимагати певної послідовності.

Специфікація діяльності (на вищих рівнях сумісності) може дозволяти виконання декількох (логічних) потоків, та існування механізмів синхронізації для гарантування виконання дій у правильному порядку.

Діаграма прецедентів це діаграма, на якій зображено відношення між акторами та прецедентами в системі. Також, перекладається як діаграма варіантів використання.

Діаграма прецедентів є графом, що складається з множини акторів, прецедентів (варіантів використання) обмежених границею системи (прямокутник), асоціацій між акторами та прецедентами, відношень серед прецедентів, та відношень узагальнення між акторами. Діаграми прецедентів відображають елементи моделі варіантів використання.

Суть даної діаграми полягає в наступному: проєктована система представляється у вигляді безлічі сутностей чи акторів, що взаємодіють із системою за допомогою так званих варіантів використання. Варіант використання (use case) використовують для описання послуг, які система надає актору. Іншими словами, кожен варіант використання визначає деякий набір дій, який виконує система при діалозі з актором.

При цьому нічого не говориться про те, яким чином буде реалізована взаємодія акторів із системою.

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		51

У мові UML є кілька стандартних видів відношень між акторами і варіантами використання:

- асоціації (association relationship);
- включення (include relationship);
- розширення (extend relationship);
- узагальнення (generalization relationship).

При цьому загальні властивості варіантів використання можуть бути представлені трьома різними способами, а саме – за допомогою відношень включення, розширення і узагальнення.

Відношення асоціації – одне з фундаментальних понять у мові UML і в тій чи іншій мірі використовується при побудові всіх графічних моделей систем у формі канонічних діаграм.

Включення (include) у мові UML – це різновид відношення залежності між базовим варіантом використання і його спеціальним випадком. При цьому відношенням залежності (dependency) є таке відношення між двома елементами моделі, при якому зміна одного елемента (незалежного) приводить до зміни іншого елемента (залежного).

Відношення розширення (extend) визначає взаємозв'язок базового варіанта використання з іншим варіантом використання, функціональна поведінка якого задіюється базовим не завжди, а тільки при виконанні додаткових умов.

Діаграма класів це статичне представлення структури моделі. Відображає статичні (декларативні) елементи, такі як: класи, типи даних, їх зміст та відношення.

Діаграма класів, також, може містити позначення для пакетів та може містити позначення для вкладених пакетів. Також, діаграма класів може містити позначення деяких елементів поведінки, однак їх динаміка розкривається в інших типах діаграм.

Діаграма класів (class diagram) служить для представлення статичної структури моделі системи в термінології класів об'єктно-орієнтованого

програмування. На цій діаграмі показують класи, інтерфейси, об'єкти й кооперації, а також їхні відносини.

В UML існують наступні типи зв'язків які використовуються у діаграмі класів: Асоціації; Агрегація; Композиція.

Асоціації це якщо між двома класами визначена асоціація, то можна переміщатися від об'єктів одного класу до об'єктів іншого. Цілком припустимі випадки, коли обидва кінці асоціації відносяться до одного і того ж класу. Це означає, що з об'єктом деякого класу дозволено зв'язати інші об'єкти з того ж класу. Асоціація, що зв'язує два класи, називається бінарної. Можна, хоча це рідко буває необхідним, створювати асоціації, що зв'язують відразу кілька класів. Графічно асоціація зображується у вигляді лінії, що з'єднує клас сам з собою або з іншими класами.

Асоціації може бути присвоєно ім'я, яке описує природу відносини. Зазвичай ім'я асоціації не вказується, якщо тільки ви не хочете явно задати для неї рольові імена або у вашій моделі настільки багато асоціацій, що виникає необхідність посилатися на них і відрізняти один від одного. Ім'я буде особливо корисним, якщо між одними і тими ж класами існує кілька різних асоціацій.

Клас, що бере участь в асоціації, грає в ній деяку роль. По суті, це "обличчя", яким клас, що знаходиться на одній стороні асоціації, звернений до класу з іншого її боку. Можна явно позначити роль, яку клас грає в асоціації.

Часто при моделюванні буває важливо вказати, скільки об'єктів може бути пов'язано допомогою одного примірника асоціації. Це число називається кратністю (Multiplicity) ролі асоціації та записується або як вираз, значенням якого є діапазон значень, або в явному вигляді.

Вказуючи кратність на одному кінці асоціації, ви тим самим говорите, що на цьому кінці саме стільки об'єктів повинно відповідати кожному об'єкту на протилежному кінці. Кратність можна задати рівною одиниці (1), можна вказати діапазон: "нуль або одиниця" (0..1), "багато" (0.. *), "одиниця або більше" (1.. *). Дозволяється також вказувати певне число (наприклад, 3). За допомогою списку

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		53

можна задати і більш складні кратності, наприклад 0.. 1, 3..4, 6.. *, що означає "будь-яке число об'єктів, крім 2 і 5".

Агрегація це проста асоціація між двома класами відображає структурний відношення між рівноправними сутностями, коли обидва класу знаходяться на одному концептуальному рівні і ні один не є більш важливим, ніж інший. Але іноді доводиться моделювати відношення типу «частина/ціле», в якому один з класів має більш високий ранг (ціле) і складається з декількох менших за рангом (частин).

Ставлення такого типу називають агрегацією; воно зараховане до відносин типу «має» (з урахуванням того, що об'єкт-ціле має кілька об'єктів-частин). Агрегація є окремим випадком асоціації і зображується у вигляді простої асоціації з незафарбованим ромбом з боку «цілого». Графічно агрегація представляється порожнім ромбом на блоці класу, і лінією, яка від цього ромба до міститься класу.

Композиція це більш суворий варіант агрегації. Відома також як агрегація за значенням.

Композиція має жорстку залежність часу існування екземплярів класу контейнера та примірників містяться класів. Якщо контейнер буде знищений, то весь його вміст буде також знищено. Графічно представляється як і агрегація, але з зафарбованим ромбиком.

Діаграма компонент в UML це діаграма, на якій відображаються компоненти, залежності та зв'язки між ними.

Діаграма компонент відображає залежності між компонентами програмного забезпечення, включаючи компоненти вихідних кодів, бінарні компоненти, та компоненти, що можуть виконуватись.

Модуль програмного забезпечення може бути представлено в якості компоненти. Деякі компоненти існують під час компіляції, деякі – під час компонування, а деякі під час роботи програми.

Діаграма компонент відображає лише структурні характеристики, для відображення окремих екземплярів компонент слід використовувати діаграму розгортання.

Компоненти об'єднуються разом використовуючи структурні зв'язки (assembly connector) щоб об'єднати інтерфейси двох компонент. Це ілюструє зв'язок типу «клієнт-сервер».

Структурна взаємодія – «зв'язок двох компонент, який передбачає, що один з них надає послуги, потрібні іншому компоненту».

При використанні діаграми компонент щоб показати внутрішню структуру компонента, клієнтські та серверні інтерфейси можуть утворювати пряме з'єднання з внутрішніми. Таке з'єднання називається з'єднанням делегації.

Діаграма об'єктів в UML це діаграма, що відображає об'єкти та їх зв'язки в певний момент часу. Діаграма об'єктів може розглядатись як окремий випадок діаграми класів, на якій можуть бути представлені як класи, так і екземпляри (об'єкти) класів. Схожою за змістом є діаграма взаємодії (collaboration diagram).

Діаграми об'єктів не мають власної нотації. Оскільки діаграми класів можуть відображати об'єкти, то діаграма класів, на якій відображено лише об'єкти, та не відображено класи, може вважатись діаграмою об'єктів.

Діаграма об'єктів відображає об'єкти та зв'язки в певний момент роботи програми. Об'єкти можуть містити інформацію про власні значення а не про описання. Для відображення загальних шаблонів об'єктів та зв'язків, що можуть багаторазово створюватись під час роботи програми, слід використовувати діаграму взаємодії, яка може відображати характеристики об'єктів та зв'язків. Екземпляр діаграми взаємодії створює діаграму об'єктів.

Діаграма об'єктів не відображає еволюцію системи під час роботи. Натомість, слід використовувати діаграми взаємодії з повідомленнями, або діаграми послідовності.

Діаграма розгортання (deployment diagram) це діаграма в UML, на якій відображаються обчислювальні вузли під час роботи програми, компоненти, та

об'єкти, що виконуються на цих вузлах. Компоненти відповідають представленню робочих екземплярів одиниць коду. Компоненти, що не мають представлення під час роботи програми на таких діаграмах не відображаються; натомість, їх можна відобразити на діаграмах компонент. Діаграма розгортання відображає робочі екземпляри компонент, а діаграма компонент, натомість, відображає зв'язки між типами компонент.

Опис розробленої системи

Програмне забезпечення системи інтелектуального моніторингу та оптимізації ІТ систем ЦОД працює як прикладна Python система для збирання, аналізу та оцінювання стану серверів, мережевого обладнання, сховищ даних і сервісів дата центру.

Система отримує показники завантаження процесора, оперативної пам'яті, дисків, мережевого трафіку, температури та доступності вузлів.

На основі цих даних програма визначає перевантажені ресурси, формує попередження, розраховує інтегральний індекс стану ЦОД і пропонує дії для оптимізації.

Архітектура системи складається з модуля збору метрик, модуля зберігання даних, аналітичного модуля, модуля інтелектуальної оцінки, модуля рекомендацій, модуля звітності та графічного або веб інтерфейсу адміністратора.

Основні функції системи:

1. Збір технічних параметрів вузлів ЦОД.
2. Збереження історії моніторингу у базі даних SQLite.
3. Виявлення перевантаження серверів і сервісів.
4. Розрахунок рівня ризику для кожного вузла.
5. Формування рекомендацій щодо оптимізації ресурсів.
6. Створення звітів для адміністратора.
7. Відображення поточного стану інфраструктури ЦОД.

В цілому система допомагає адміністратору швидко оцінювати стан інфраструктури, виявляти проблемні вузли, планувати перерозподіл

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		56

навантаження, контролювати стабільність сервісів і зменшувати ризик простою ІТ систем ЦОД.

Розглянемо фрагмент реалізованого вихідного коду де представлено структуру.

```
import sqlite3
import random
from datetime import datetime
class DataCenterMonitor:
    def __init__(self, db_name="datacenter_monitor.db"):
        self.db_name = db_name
        self.create_tables()
    def create_tables(self):
        #Створення таблиці для збереження результатів моніторингу
        with sqlite3.connect(self.db_name) as connection:
            cursor = connection.cursor()
            cursor.execute("""
                CREATE TABLE IF NOT EXISTS monitoring_data (
                    id INTEGER PRIMARY KEY AUTOINCREMENT,
                    node_name TEXT NOT NULL,
                    cpu_load REAL NOT NULL,
                    ram_load REAL NOT NULL,
                    disk_load REAL NOT NULL,
                    network_load REAL NOT NULL,
                    temperature REAL NOT NULL,
                    risk_level TEXT NOT NULL,
                    recommendation TEXT NOT NULL,
                    created_at TEXT NOT NULL
                )
            """)
            connection.commit()
    def collect_metrics(self, node_name):
        #Імітація збору метрик з вузла ЦОД
        return {
            "node_name": node_name,
            "cpu_load": round(random.uniform(20, 98), 2),
            "ram_load": round(random.uniform(25, 96), 2),
            "disk_load": round(random.uniform(30, 99), 2),
            "network_load": round(random.uniform(10, 95), 2),
            "temperature": round(random.uniform(28, 75), 2)
        }
```

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		57

```

def analyze_metrics(self, metrics):
#Інтелектуальна оцінка стану вузла на основі критичних параметрів
    risk_score = 0
    if metrics["cpu_load"] > 85:
        risk_score += 2
    if metrics["ram_load"] > 85:
        risk_score += 2
    if metrics["disk_load"] > 90:
        risk_score += 2
    if metrics["network_load"] > 80:
        risk_score += 1
    if metrics["temperature"] > 65:
        risk_score += 3
    if risk_score >= 6:
        return "Критичний"
    if risk_score >= 3:
        return "Підвищений"
    return "Нормальний"

def generate_recommendation(self, metrics, risk_level):
#Формування практичної рекомендації для адміністратора ЦОД
    if risk_level == "Критичний":
        if metrics["temperature"] > 65:
            return "Перевірити охолодження серверної стійки
та зменшити навантаження на вузол"
        if metrics["disk_load"] > 90:
            return "Очистити сховище, перенести частину даних
або збільшити дисковий ресурс"
        return "Перерозподілити навантаження між серверами
та перевірити активні сервіси"
    if risk_level == "Підвищений":
        return "Провести профілактичний аналіз навантаження
та перевірити резерв ресурсів"

    return "Стан вузла стабільний, оптимізація не потрібна"

def save_result(self, metrics, risk_level, recommendation):
#Збереження результатів аналізу у базі даних
    with sqlite3.connect(self.db_name) as connection:
        cursor = connection.cursor()
        cursor.execute("""
            INSERT INTO monitoring_data (
                node_name, cpu_load, ram_load, disk_load,
                network_load, temperature, risk_level,

```

```

        recommendation, created_at
    )
    VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)
"""
    (
        metrics["node_name"],
        metrics["cpu_load"],
        metrics["ram_load"],
        metrics["disk_load"],
        metrics["network_load"],
        metrics["temperature"],
        risk_level,
        recommendation,
        datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    ))
    connection.commit()

def monitor_node(self, node_name):
    metrics = self.collect_metrics(node_name)
    risk_level = self.analyze_metrics(metrics)
    recommendation = self.generate_recommendation(metrics, risk_level)
    self.save_result(metrics, risk_level, recommendation)
    print("Вузол:", metrics["node_name"])
    print("CPU:", metrics["cpu_load"], "%")
    print("RAM:", metrics["ram_load"], "%")
    print("Диск:", metrics["disk_load"], "%")
    print("Мережа:", metrics["network_load"], "%")
    print("Температура:", metrics["temperature"], "C")
    print("Рівень ризику:", risk_level)
    print("Рекомендація:", recommendation)

def main():
    monitor = DataCenterMonitor()
    nodes = ["Server 01", "Server 02", "Storage 01", "Router 01"]
    for node in nodes:
        monitor.monitor_node(node)
        print()

if __name__ == "__main__":
    main()

```

4.2 Захист розробленого програмного забезпечення

Для захисту розробленого програмного забезпечення запропоновано використовувати алгоритм SHACAL-1, який заснований на перетвореннях алгоритму SHA-1. SHACAL-1 шифрує 160-бітний блок даних з використанням 512-бітного ключа шифрування. Допускається використання більше коротких ключів шифрування (не коротше 128 біт), які перед виконанням розширення ключа (процедура розширення ключа також успадкована від SHA-1 і буде описана нижче) повинні бути доповнені нульовими бітами для досягнення 512-бітного розміру.

В алгоритмі SHACAL-1 передбачено 80 раундів шифрування. Шифруєме повідомлення представляється у вигляді п'яти 32-бітних субблоків A , B , C , D і E , над якими в кожному раунді виконуються наступні дії:

$$A_{i+1} = K_i + (A_i \lll 5) + f_i(B_i, C_i, D_i) + E_i + M_i,$$

$$B_{i+1} = A_i,$$

$$C_{i+1} = B_i \lll 30,$$

$$D_{i+1} = C_i,$$

$$E_{i+1} = D_i,$$

де i – номер раунду ($i = 0 \dots 79$),

K_i – фрагмент розширеного ключа для i -го раунду,

f_i – функція для i -го раунду (див. нижче),

$\lll$ – операція побітового циклічного зрушення вліво,

M_i – константи, що модифікують, певні в такий спосіб:

Раунди	Значення константи
0...19	5A827999
20...39	6ED9EBA1
40...59	8F1BBCDC
60...79	CA62C1D6

Використовувані в раундах функції f_i визначені так:

Раунди	Функція
0...19	$f(x,y,z)=(x\&y) (x'\&z)$
20...39,60...79	$f(x,y,z)=x \square \oplus y \square \oplus z$
40...59	$f(x,y,z)=(x \square \oplus y) (x \oplus z) (y \oplus z)$

У таблиці символами $\&$, $|$ і $\oplus$ позначені, відповідно, побітові логічні операції «і», «або» й «або, що виключає» (XOR); x' позначає побітовий комплемент до x .

Шифртекстом є конкатенація вмісту змінних A_{80} , B_{80} , C_{80} , D_{80} і E_{80} .

Процедура розширення ключа в алгоритмі SHACAL-1 також досить проста, вона виконується у два етапи:

Етап 1. 512-бітний вихідний ключ шифрування ділиться на 16 фрагментів по 32 біта $K_0 \dots K_{15} \dots$

Етап 2. Інші фрагменти розширеного ключа $K_{16} \dots K_{79}$ обчислюються з перших 16 фрагментів у такий спосіб:

$$K_i = (K_{i-3} \oplus K_{i-8} \oplus K_{i-14} \oplus K_{i-16}) \lll 1.$$

Раунди розшифрування виконуються у зворотній послідовності; кожний з них має на увазі виконання наступних операцій:

$$A_i = B_{i+1},$$

$$B_i = C_{i+1} \lll 2,$$

$$C_i = D_{i+1},$$

$$D_i = E_{i+1},$$

$$E_i = K'_i + (B_{i+1} \lll 5)' + f'_i(C_{i+1} \lll 2, D_{i+1}, E_{i+1}) + A_{i+1} + M'_i + 4.$$

Тут запис $f(x)$ позначає побітовий комплемент результату виконання операції $f(x)$.

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

На рисунку 5.1 зображено інтерфейс програмного забезпечення, розробленого у результаті виконання бакалаврської дипломної роботи.

Розроблене програмне забезпечення системи моніторингу та оптимізації ІТ-систем ЦОД складається з наступних функціональних блоків:

- Блоку виведення поточного трафіку.
- Блоку журналу роботи ПЗ.
- Блоку виведення списку ПК.
- Блоку виведення IP-адрес ПК та побудови дерева ресурсів.
- Навігаційного меню яке викликається натисканням правої клавіші маніпулятора миші.
- Функціональних кнопок ПЗ.

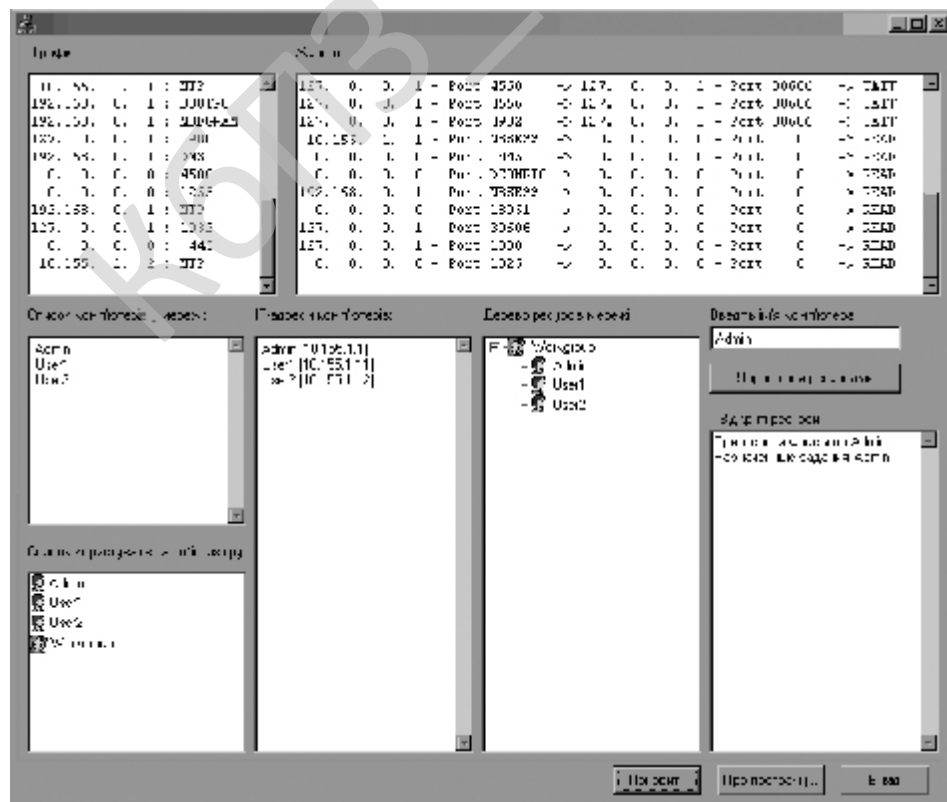


Рисунок 5.1 – Головне вікно розробленого ПЗ

Для перегляду короткої довідки про програму слід натиснути на основному вікні кнопку авторського права, після чого на екрані з'явиться вікно показане на рисунку 5.2.

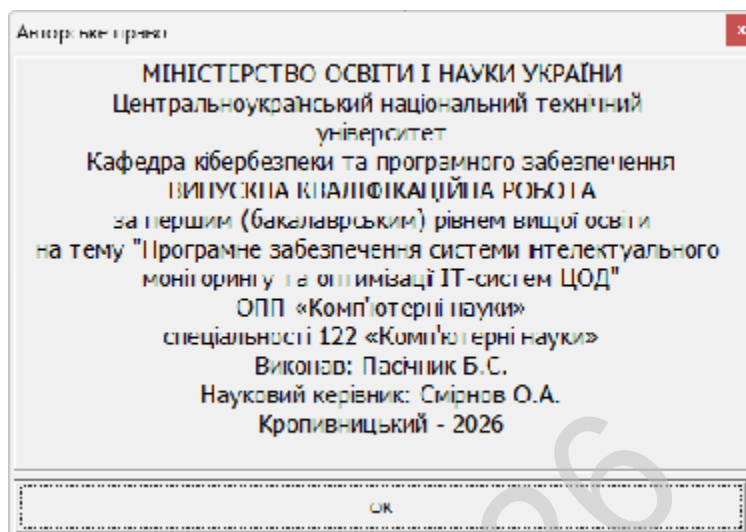


Рисунок 5.2 – Вікно розробника ПЗ

Під час роботи над програмою було проведено тестування програмного забезпечення, тобто технічне дослідження, призначене для виявлення інформації про якість продукту відносно контексту, в якому воно має використовуватись.

Тестування включає як процес пошуку помилок або інших дефектів, так і випробування програмних складових з метою їх оцінки.

Проводилась оцінка:

- відповідності поставленим вимогам;
- правильна відповідь для усіх можливих вхідних даних;
- виконання функцій за прийнятний час;
- практичність;
- сумісність з ОС та стороннім ПЗ.

Оскільки число можливих тестів для програмних компонент практично нескінченне, тому стратегія тестування полягала в тому, щоб провести всі можливі тести з урахуванням наявного часу та ресурсів.

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		63

Як результат ПЗ тестувалось стандартним виконанням програми з метою виявлення помилок або інших дефектів.

Проводилось тестування форматом чорної скриньки. Основне місце програми тестів «чорної скриньки» – інтерфейс ПЗ. Відомі: функції програми. Досліджується: робота кожної функції на всій області визначення.

Ці тести демонструють:

- Як виконуються функції програми.
- Як приймаються вихідні дані.
- Як виробляються результати.
- Як зберігається цілісність зовнішньої інформації.

При тестуванні «чорної скриньки» розглядаються системні характеристики програм, ігнорується їхня внутрішня логічна структура. Вичерпне тестування, як правило, неможливе.

Наприклад, якщо в програмі 10 вхідних величин і кожна приймає по 10 значень, то кількість тестових варіантів становитиме 10^{10} . Тестування «чорної скриньки» не реагує на багато особливостей програмних помилок.

Тестування «чорної скриньки» (функціональне тестування) дозволяє отримати комбінації вхідних даних, які забезпечують повну перевірку всіх функціональних вимог до програми.

Програмний виріб тут розглядається як «чорна скринька», чию поведінку можна визначити тільки дослідженням його входів та відповідних виходів. При такому підході бажано мати:

- Набір, утворений такими вхідними даними, які призводять до аномалій у поведінці програми (назвемо його ІТс).
- Набір, утворений такими вхідними даними, які демонструють дефекти програми (назвемо його ОТ).

Будь-який спосіб тестування «чорної скриньки» повинен:

- Виявити такі вхідні дані, які з високою ймовірністю належать набору ІТс.

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		64

– Сформулювати такі очікувані результати, які з високою імовірністю є елементами набору ОТ.

Принцип «чорної скриньки» не альтернативний принципу «білої скриньки». Скоріше це доповнює підхід, який виявляє інший клас помилок. Тестування «чорної скриньки» забезпечує пошук наступних категорій помилок:

- Некоректних чи відсутніх функцій;
- помилок інтерфейсу;
- помилок у зовнішніх структурах даних або в доступі до зовнішньої бази даних;
- помилок характеристик (необхідна ємність пам'яті і т.д.);
- помилок ініціалізації та завершення.

Обрано умови розповсюдження – Shareware. Під умовно-безплатним програмним забезпеченням можна розуміти спосіб або метод розповсюдження комерційного ПЗ на ринку (тобто на шляху до кінцевого користувача), при якому випробувачеві пропонується обмежена за можливостями (не повнофункціональна або демонстраційна версія), терміном дії (тріал версія) або версія з вбудованим набридливим нагадуванням про необхідність оплати використання програми. В угоді про використання (ліцензії для кінцевого користувача, EULA) також може бути обумовлена заборона на комерційне або професійне (не тестове) її використання. Основний принцип умовно-безплатного ПЗ – «спробуй, перш ніж купити» (try before you buy). ПЗ що поширюється як умовно-безплатний, надається користувачам безоплатно. Звичайно користувач платить тільки за час завантаження файлів через Інтернет або за носій (CD диск, флешку, ключ). Протягом певного терміну, що становить зазвичай тридцять днів, він може користуватися програмою, тестувати її, освоювати її можливості. Якщо після закінчення цього терміну користувач вирішить продовжити використання ПЗ, він зобов'язаний купити його (zareєstrуватися), заплативши авторові певну суму. В іншому випадку користувач повинен припинити використання ПЗ та видалити його зі свого комп'ютера.

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		65

6 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти, призначено для системи інтелектуального моніторингу та оптимізації ІТ-систем ЦОД.

В межах України в недостатній мірі представлені вітчизняні розробки в цій області.

Рішення завдання полягало у вирішенні наступних задач:

- Був проведений огляд існуючих систем інтелектуального моніторингу та оптимізації ІТ-систем ЦОД.
- Досліджена система інтелектуального моніторингу та оптимізації ІТ-систем ЦОД.
- На основі отриманих результатів досліджень створена програмна реалізація системи інтелектуального моніторингу та оптимізації ІТ-систем ЦОД.

Розроблені під час виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання інтелектуального моніторингу та оптимізації ІТ-систем ЦОД.

Розроблене програмне забезпечення має простий, дружній та зручний інтерфейс користувача, що забезпечує легкість у освоєнні роботи програмного продукту, зручність у використанні, і не потребує особливих спеціальних знань.

При створенні програмного забезпечення було використано об'єктно-орієнтований підхід, що відповідає сучасним тенденціям у галузі розробки комерційних програмних систем.

Програма реалізована на мові високого рівня Python. Дана мова програмування дозволяє найбільш ефективно обробляти дані призначені для системи інтелектуального моніторингу та оптимізації ІТ-систем ЦОД. Це дозволило мінімізувати строк розробки програмного забезпечення, і, як слід,

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		66

зменшити витрати на його розробку. Запропоноване програмне забезпечення ділиться на загальне програмне забезпечення, що поставляється із засобами обчислювальної техніки й спеціальне програмне забезпечення, що спеціально розроблене для даної конкретної системи й включає програми, що реалізують її функції.

Програма призначена для виконання під управлінням багатозадачної операційної системи Windows 10/11.

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

Для підвищення рівня безпеки запропоновано застосовувати алгоритм SHACAL-1.

В цілому створене програмне забезпечення підтверджує правильність використаних проектних рішень та повністю відповідає вимогам технічного завдання. Створене програмне забезпечення має потенційну можливість для подальшого вдосконалення і застосування у різних галузях.

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		67

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Doug Lowe «Networking For Dummies 12th Edition». 2020. – 480 p.
2. Ramon Nastase «Computer Networking: The Beginner's guide for Mastering Computer Networking, the Internet and the OSI Model». 2018. – 186 p.
3. Russ White & Ethan Banks «Computer Networking Problems and Solutions: An Innovative Approach to Building Resilient, Modern Networks». 2017. – 832 p.
4. Вінтенко Б., Смірнов О., Миронець І., Смірнова Т., Смірнов С. «Імітаційна модель шляхів вхідних даних комп'ютерної інтелектуальної системи підтримки оператора енергоблоку АЕС». *Комбінаторні конфігурації та їхні застосування: Матеріали XXVII Міжнародного науково-практичного семінару, присвяченого 125-річчю Національного університету «Запорізька політехніка»* (Запоріжжя-Кропивницький-Київ, 4-6 червня 2025 р.). Запоріжжя: НУ «Запорізька політехніка», 2025. С.82-91.
5. Al-Azzeh, J., Ayyoub, B., Mesleh, A., Smirnova, T., Gnatyuk, S., Drieiev, O., Smirnov, O., Dorenskyi, O. «Cloud-Based Information System for Evaluating Caverns in the Process of Blasting Metal Surfaces of Details». *International Review on Modelling and Simulations* 18 (1), 2025. pp. 32-42.
6. Смірнова Т.В., Коноплицька-Слободенюк О.К., Буравченко К.О., Смірнов С.А., Кравчук О.В., Козірова Н.Л., Смірнов О.А. «Дослідження технологій забезпечення кібербезпеки хмарних сервісів IaaS, PaaS та SaaS». *Кібербезпека: освіта, наука, техніка*. 2024. №4(24), С. 6-27.
7. Батрак О., Смірнова Т., Гнатюк В., Одарченко Р., Смірнов О. «Дослідження показників ефективності функціонування та перспектив розвитку систем IP-телефонії». *Підводні технології*, 2024, № 13, с. 28-35.
8. Kuznetsov, O., Kryvinska, N., Ilchenko, O., Smirnova, T., Ulianovska, Y. «Comparative Analysis of Cryptocurrency Trading Platforms Using the Analytic Hierarchy Process». *CEUR Workshop Proceedings*, 2023, 3628, pp. 106-115.

9. Al-Mudhafar Aqeel, A.M., Smirnova, T., Buravchenko, K., Smirnov, O. «The method of assessing and improving the user experience of subscribers in software-configured networks based on the use of machine learning». *Advanced Information Systems*, 2023, 7(2), pp. 49-56.

10. Smirnov, O., Sydorenko, V., Aleksander, M., Zhyharevych, O., Yenchев, S. «Simulation of the cloud IoT-based monitoring system for critical infrastructures». *CEUR Workshop Proceedings*, Volume 3530, 2023, pp. 256-265.

11. Smirnov, O., Odarchenko, R., Smirnova, T., Bondar, S., Volosheniuk, D. «Optimal Structure Construction of Private 5G Network for the Needs of Enterprises». *Lecture Notes on Data Engineering and Communications Technologies*, 2023, 178, pp. 208–223.

12. Аль-Мудхафар Акіл Абдулхуссейн М., Смірнова Т.В., Буравченко К.О., Смірнов О.А. «Метод оцінки та підвищення користувальницького досвіду абонентів в програмно-конфігурованих мережах на основі використання машинного навчання». *Сучасні інформаційні системи*, 2023, том 7, № 2, С. 49-56.

13. Smirnova, T., Gnatyuk, S., Yudin, O., Sydorenko, V., Polozhentsev, A., «The Model for Calculating the Quantitative Criteria for Assessing the Security Level of Information and Telecommunication Systems». *CEUR Workshop Proceedings* Volume 3156, 2022, Pages 390-399.

14. Смірнова Т.В., Гнатюк С.О., Сидоренко В.М., Юдін О.Ю., Сидоренко С.Ю., «Модель визначення критичності галузевих інформаційно-телекомунікаційних систем». *Проблеми інформатизації та управління*, № 2(70). 2022. С. 28-37.

15. Смірнов О.А., Смірнова Т.В., Якименко Н.М., Смірнов С.А., Поліщук Л.І., «Дослідження стійкості до диференціального криптоаналізу запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Системи управління, навігації та зв'язку*, 2022, № 3(69). С. 93-98.

16. Смірнов О.А., Смірнова Т.В., Якименко Н.М., Поліщук Л.І., Смірнов С.А. «Дослідження статистичної стійкості та швидкісних характеристик запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Вісник Хмельницького національного університету. Серія: «Технічні науки»*, № 2 (307). С. 46-52. 2022.

17. Смірнов О.А., Смірнова Т.В., Константинова Л.В., Смірнов С.А., Якименко Н.М., «Дослідження стійкості до лінійного криптоаналізу запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Системи управління, навігації та зв'язку*, 2022, № 1(67). С. 84-89.

18. Smirnova T., Gnatyuk S., Berdibayev R., Avkurova Zh., Iavich M. «Cloud-Based Cyber Incidents Response System and Software Tools». *Communications in Computer and Information Science*, 2021, vol 1486. Springer, Cham. pp 169-184.

19. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova T. «Non-binary constant weight coding technique». *CEUR Workshop Proceedings*. Volume 2740, 2020, Pages 102-114.

20. Smirnov O., Alimseitova Zh., Adranova A., Akhmetov B., Lakhno V., Zhilkishbayeva G. «Models and algorithms for ensuring functional stability and cybersecurity of virtual cloud resources». *Journal of theoretical and applied information technology* Vol.98. No 21, 2020, P. 3334-3346.

21. Smirnov O., Kuznetsov A., Kiian A., Cherep A., Kanabekova M., Chepurko I. «Testing of code-based pseudorandom number generators for post-quantum application». *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, Ukraine, Kyiv, May 14-18. 2020. P. 172-177.

22. Smirnov O., Kuznetsov A., Pushkar'ov A., Serhiienko R., Babenko V., Kuznetsova T., «Representation of Cascade Codes in the Frequency Domain». In: Radivilova T., Ageyev D., Kryvinska N. (eds) *Data-Centric Business and*

Applications. Lecture Notes on Data Engineering and Communications Technologies, vol 48. Springer, Cham. 2021. pp 557-587.

23. Smirnov, O., Markovets, O. Vovk, N., Turchyn, Y., «Model of informational support for social network administrators' content creation». *CEUR Workshop Proceedings* Volume 2616, 2020, Pages 125-136.

24. Smirnov, O., Drieieva, H., Drieiev, O., Polishchuk, Y., Brzhanov, R., Aleksander, M. «Method of fractal traffic generation by a model of generator on the graph». *CEUR Workshop Proceedings* Volume 2616, 2020, Pages 366-379.

25. Smirnov, O., Drieieva, H., Drieiev, O., Simakhin, V., Bondar, S., Odarchenko, R. «Managing multifractal properties of the binary sequence generated with the Markov chains», *CEUR Workshop Proceedings* Volume 2608, 2020, Pages 633-645.

26. Smirnov O. Kuznetsov A., Zaichenko Yu., Pastukhov M., Oleshko O., Kuznetsova K., «Formation of Discrete Signals with Special Correlation Properties». *International Conference on Information and Telecommunication Technologies and Radio Electronics, UkrMiCo 2019*; Odessa; Ukraine; 9-13 September 2019. P.22-28.

27. Smirnov, O., Kuznetsov, A., Kolovanova, I., Kuznetsova, T., «Noise immunity of the algebraic geometric codes». *International Journal of Computing*; 2019, Volume 18, Issue 4 – Research Institute for Intelligent Computer Systems – 2019. – P. 393-407.

28. Smirnov, O., Kuznetsov, A., Reshetniak, O., Ivko, N., Katkova, T., Kuznetsova, T., «Generators of Pseudorandom Sequence with Multilevel Function of Correlation». *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T)*, Kyiv, Ukraine, 8 – 11 October 2019. P.517-522.

29. Smirnov, O., Odarchenko, R., Abakumova, A., Usik, P., Kundyz, M., «QoE optimization technique for media delivery in 5G networks». *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T)*, Kyiv, Ukraine, 8 – 11 October 2019. P.597-601.

30. Smirnov, O., Krasnobayev, V., Yanko, A., Kuznetsova, T. «Methods of nulling numbers in the system of residual classes». *CEUR Workshop Proceedings*, Vol 2588, P. 90-106, 2019.

31. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Averchev, A., Pastukhov, M., Kuznetsova, K., «Formation of Pseudorandom Sequences with Special Correlation Properties», *2019 3rd International Conference on Advanced Information and Communications Technologies, AICT -2019/ Lviv, Ukraine, 2-6 July, 2019*, P. 395-399.

32. Smirnov, O., Kuznetsov, A., Kiian, A., Zamula, A., Rudenko, S., Hryhorenko, V., «Variance Analysis of Networks Traffic for Intrusion Detection in Smart Grids», *2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS)*, Kyiv, Ukraine April 17-19, 2019 P. 353-358.

33. Smirnov, O., Kuznetsov, A., Kavun, S., Babenko, B., Nakisko, O., Kuznetsova, K., «Malware Correlation Monitoring in Computer Networks of Promising Smart Grids», *2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS)*, Kyiv, Ukraine April 17-19, 2019 P. 347-352.

34. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Pastukhov, M., Kuznetsova, K., Prokopovych-Tkachenko, D., «Discrete Signals with Special Correlation Properties», *CEUR Workshop Proceedings Volume 2353, CEUR Workshop Proceedings 2019*, Pages 618-629.

35. Smirnov A.A., Kuznetsov A.A., Danilenko D.A., Berezovsky A., «The statistical analysis of a network traffic for the intrusion detection and prevention systems», *Telecommunications and Radio Engineering*. – Volume 74, Issue 1. – Begel House Inc. – 2015. – P. 61-78.

36. Смірнов О.А., Смірнова Т.В., Буравченко К.О., Кравченко С.С., Горбов В.О., «Хмарна система підтримки прийняття рішень технологічного процесу відновлення поверхонь конструкцій і деталей машин». *Сучасні інформаційні системи*. 2021. Т. 5, № 4. С. 79-95

37. Смірнов О.А., Усік П.С., Миронець І.В., Буравченко К.О., Якименко Н.М. «Метод підвищення ефективності розподіленої обробки даних у

					ВКРБ-122.26.0060.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		72

комп'ютерних системах операторів стільникового зв'язку» *Вісник Черкаського державного технологічного університету. Технічні науки.* №4. С. 103-110. 2020.

38. О.А.Смірнов, Т.В.Смірнова, Л.І. Поліщук, К.О. Буравченко, А.О.Макевнін, «Дослідження хмарних технологій як сервісів», *Кібербезпека: освіта, наука, техніка.* № 3(7). С. 43-62. 2020.

39. Смірнов О.А., Коноплицька-Слободенюк О.К., Смірнов С.А., Буравченко К.О., Смірнова Т.В., Поліщук Л.І. Інформаційна безпека в комп'ютерних мережах. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2020. – 294 с.

40. О.А. Смірнов, П.С. Усік, «Дослідження перспектив використання технологічних рішень в мережах 5G» у *Кібербезпека та інформаційні технології: монографія.* – Х. : ТОВ «ДІСА ПЛЮС», 2020.С. 122-135.

41. Смірнов О.А., Дреєва Г.М., Дреєв О.М., Смірнова Т.В. «Фрактальний аналіз генератора самоподібного трафіку на основі ланцюга Маркова». *Центральноукраїнський науковий вісник. Технічні науки.* № 2(33). с. 161-172, 2019.

42. Смірнов О.А., Коноплицька-Слободенюк О.К., Смірнов С.А., Буравченко К.О., Смірнова Т.В. Поліщук Л.І. Проектування комп'ютерних систем та мереж. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2019. – 264 с.

43. Smirnov, O., Kuznetsov, A., Kuznetsova., K. Synthesis of Discrete Signals with Improved Correlation Properties. Монографія: In.: ISCI'2019: Information Security in Critical Infrastructures. Collective monograph. Edited by Ivan D. Gorbenko and Alexandr A. Kuznetsov, ASC Academic Publishing, USA, 2019, pp. 281-299. – ISBN: 978-0-9989826-8-7 (Hardback), ISBN: 978-0-9989826-9-4 (Ebook).

44. Смірнов О.А., Дреєва Г.М. Метод генерування фрактального трафіку за допомогою моделі генератора на графі. Монографія: Інформаційна безпека та інформаційні технології : монографія / за заг. ред. В. С. Пономаренка. – Х. : Вид. Рожко С.Г. 2019. С. 123-139

					ВКРБ-122.26.0060.00.00.ПЗ	Арк. 73
Вим.	Арк.	№ докум.	Підпис	Дата		

45. Дреєва Г.М., Смірнов О.А., Дреєв О.М. Метод генерування фрактальноподібної числової послідовності на основі скінченного автомату для моделювання трафіку у мережі. Центральнотраїнський науковий вїсник. Технїчні науки. № 1(32). с. 173-183, 2019.

46. Смірнова Т.В., Солових Є.К., Смірнов О.А., Дреєв О.М. Побудова хмарних інформаційних технологій оптимізації технологічного процесу відновлення та зміцнення поверхонь деталей. Центральнотраїнський науковий вїсник. Технїчні науки. № 1(32). с. 184-194, 2019.

47. Смірнов О.А., Смірнов С.А., Полїщук Л.І., Смірнова Т.В., Коноплицька-Слободенюк О.К. Метод формування антивірусного захисту даних з використанням безпечної маршрутизації метаданих. Кїбербезпека: освіта, наука, технїка. – Том 3 № 3. – Кїїв: КУ ім. Бориса Грінченка. – 2019. – С. 63-87.

48. Смірнов О.А., Гнатюк С.О., Кавун С.В., Терейковський І.А., Жмурко Т.О., Смірнов С.А., Коваленко А.С. Основи безпеки в комп'ютерних мережах. Навчальний посїбник – Кропивницький: вид. Лисенко В.Ф. 2018. – 177 с.

49. Смірнов О.А., Котелянець В.В. Стїйкї до колїзїй стохастичнї моделї функціонування безпроводових сенсорних мереж. Вїсник інженерної академїї України, №3, с. 145-152, 2018

50. Смірнов О.А., Смірнов С.А., Дїдик А.К., Дреєв А.М. Алгоритми формування безлїчї маршрутів передачі метаданих у антивіруснї хмарнї системи. Збїрник наукових праць "Системи обробки інформації". - Випуск 5 (142). - Х.: ХУПС - 2016. - С. 148-152.

51. Смірнов О.А., Смірнов С.А. Дїдик А.К., Дреєв О.М. Моделї системи нейромережевих експертів безпечної маршрутизації у хмарних антивірусних системах. Збїрник наукових праць "Системи обробки інформації". - Випуск 3 (140). - Х.: ХУПС - 2016. - С. 36-39.