

Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
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 мінімізації часу простою ІТ-
інфраструктури”**

Виконав здобувач вищої освіти
IV курсу, групи КІ-23-МБ
ОПП «Комп’ютерна інженерія»
спеціальності 123 «Комп’ютерна інженерія»
_____ Іванов А.В.
«___» _____ 2026 р.

Керівник проекту
кандидат фізико-математичних наук, доцент
_____ Петренюк В.І.
«___» _____ 2026 р.
Рецензент _____

Центральноукраїнський національний технічний університет
Факультет Механіко-технологічний
Кафедра Кібербезпеки та програмного забезпечення
Освітній ступінь бакалавр
Галузь знань ._12 _____“Інформаційні технології”
Спеціальність 123 “Комп’ютерна інженерія”
Освітньо-професійна (освітньо-наукова) програма “Комп’ютерна інженерія”

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

Іванову Андрію Вікторовичу

(прізвище, ім’я, по батькові)

1. Тема роботи Програмне забезпечення системи мінімізації часу простою ІТ-інфраструктури

2. Керівник роботи Петренюк Володимир Ілліч, кандидат фізико-математичних наук, доцент

(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 126-02 від 27.02.2026 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 мінімізації часу простою ІТ-інфраструктури

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

Петренюк В.І.
(прізвище та ініціали)

Завдання прийнято до виконання
« 27 » лютого 2026 р.

Підпис здобувача

Іванов А.В.
(прізвище та ініціали)

АНОТАЦІЯ

Іванов А.В. Програмне забезпечення системи мінімізації часу простою ІТ-інфраструктури. 123 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2026.

В даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи мінімізації часу простою ІТ-інфраструктури.

Метою розробки є програмне забезпечення системи мінімізації часу простою ІТ-інфраструктури.

Результат роботи – програмна реалізація системи мінімізації часу простою ІТ-інфраструктури.

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'ютерна інженерія, ІТ-інфраструктура

ABSTRACT

Ivanov A.V. Software for minimizing downtime of IT infrastructure. 123 Computer Engineering. 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 for minimizing downtime of IT infrastructure.

The purpose of the development is software for minimizing downtime of IT infrastructure.

The result of the work is software implementation of the system for minimizing downtime of IT infrastructure.

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 Windows 10/11 OS.

The program was developed in the Python environment.

Keywords: computer engineering, IT infrastructure

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	7
2.1 Огляд існуючих систем, технологій, архітектур та програмних рішень за профілем теми випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти.....	7
2.2 Обґрунтування вибору засобів для побудови системи та мови програмування.....	11
2.3 Розгорнута постановка завдання	16
3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ	17
3.1 Опис функціонування системи	17
3.2 Розробка структурної схеми.....	19
3.3 Розробка функціональної схеми	24
3.4 Розробка діаграми процесів.....	28
4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ.....	30
4.1 Розробка блок-схем та опис алгоритмів функціонування системи.....	30
4.2 Захист розробленого програмного забезпечення.....	50
5 ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ	52
6 ОСНОВНІ ВИСНОВКИ.....	54
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	56

					ВКРБ-123.26.0039.00.00.ПЗ			
Вим.	Арк.	№ докум.	Підп.	Дата	Програмне забезпечення системи мінімізації часу простою ІТ-інфраструктури	Літ.	Аркуш	Аркушів
Розроб.	Іванов А.В.					Б	1	63
Перев.	Петренко В.І.							
Н.контр.	Коваленко А.С.					ЦНТУ КІ-23-МБ		
Затв.	Смірнов О.А.							

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

ДПТМ	—	динамічний перерозподіл трафіку мережі
ЕЦП	—	електронний цифровий підпис
ІВК	—	інфраструктура відкритих ключів
ІТ	—	інформаційні технології
ЛОМ	—	локальна обчислювальна мережа
ПЗ	—	програмне забезпечення
СКЗІ	—	система комплексного захисту інформації
УК	—	управляючі компоненти
УЦ	—	удостовірюючий центр
IGMP	—	Internet Group Management Protocol
NLB	—	Network Load Balancing, балансування мережного навантаження
PKI	—	Public Key Infrastructure
VPN	—	віртуальна приватна мережа

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		2

ВСТУП

Актуальність теми. Розроблювальна в даній роботі програмне забезпечення системи мінімізації часу простою ІТ-інфраструктури за допомогою інструмента для моніторингу мережі ITInfrastructure у парі з користувацькими мережними сенсорами дозволяє запобігати несправностям до того, як вони починали турбувати користувачів.

Надійність мережі є критично важливим елементом для успішної роботи компанії. Інструмент ITInfrastructure дозволив створити десятки користувацьких мережних сенсорів, які допомагають йому передбачити більшість збоїв у роботі пристроїв до того, як вони зможуть вплинути на працездатність комунікаційної радіосистеми й нанести яка-небудь шкода діяльності його клієнтів. Програмне забезпечення для моніторингу роботи мережі ITInfrastructure допомогло скоротити час простою системи. Наші дії стали більш проактивними, ніж реактивними, тому що тепер ми можемо бачити, коли пристрій починає виходити з ладу. Це дозволяє нам у більшості випадків починати застосунок несправностей, вносити необхідні зміни, а також підготуватися до заміни пристрою, якщо це необхідно, до моменту повної відмови устаткування.

Ми здійснюємо моніторинг у режимі реального часу в нашому мережному операційному центрі, а також одержуємо оповіщення по електронній пошті й мобільні повідомлення від пристроїв. За допомогою ITInfrastructure ми можемо бачити, коли з'являється проблема, яка може виявити безпосередній негативний вплив на клієнтів ще до того, як вони встигнуть що-небудь помітити. Зараз доступ до цієї інформації став для нас набагато більш легким, що значно спростило нашу роботу.

Мета й завдання дослідження. Метою роботи є програмне забезпечення системи мінімізації часу простою ІТ-інфраструктури.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		3

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

- Огляд існуючих систем мінімізації часу простою ІТ-інфраструктури.
- Дослідження системи мінімізації часу простою ІТ-інфраструктури.
- Програмна реалізація системи мінімізації часу простою ІТ-інфраструктури.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі мінімізації часу простою ІТ-інфраструктури.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи мінімізації часу простою ІТ-інфраструктури, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		4

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Багато наших пристроїв не можуть підключатися до системи як звичайний персональний комп'ютер. Тому написання користувацьких мережних сенсорів дозволило нам контролювати роботу цих пристроїв у польових умовах.

Пристрої можуть мати більш одного сенсора, щоб допомогти нам краще зрозуміти, що із цим пристроєм відбувається. Ми можемо відслідковувати такі речі, як завантаження процесора, використання пам'яті або мережний трафік. Мої користувацькі сенсори також контролюють напругу, температуру й рівень сигналу.

Потрібно зрозуміти, як стандартний інтернет-протокол SNMP для керування пристроями в IP-мережах працює із цим конкретним програмним забезпеченням, а потім перейти в розширену віртуальну базу даних MIB (management information base, база керуючої інформації), використовувану для керування об'єктами в мережі зв'язку, щоб одержати цю інформацію.

ITInfrastructure використовує протокол SNMP, утиліту Ping, а також ETH, DNS і HTTP/HTTPS для моніторингу пристроїв у мережі. У цей час ми використовуємо VPN-тунелі із шифруванням AES, що йдуть як до стаціонарних об'єктів, так і до пристроїв з мобільним підключенням. Що стосується мобільного підключення об'єктів, то ми використовуємо укріплений мобільний стільниковий маршрутизатор для створення наших віртуальних приватних мереж (Virtual Private Network, VPN). Ці тунелі створюються тільки при включенні мобільних пристроїв. У мене також є кілька користувацьких сенсорів для цих пристроїв, які показують такі речі, що як тече стан стільникового з'єднання рівень, що тече, сигналу стільникового з'єднання, загальний місячний трафік і температура пристрою. Усі дані ми одержуємо в режимі реального часу, і вони надходять через VPN-тунелі. Ми розробили веб-портали зі змінюваною в режимі реального часу графікою, тому деякі наші клієнти можуть входити в систему й переглядати

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		5

поточну й архівну інформацію про свої власні пристрої, а також про устаткування їх об'єкта. Це дозволяє нам підвищити прозорість у відносинах з нашими клієнтами.

1.2 Область застосування

Найбільшою проблемою, з якої нам доводиться мати постійна справа, є необхідність залишатися в рамках урядових вказівок і правил безпеки. Насамперед, ми намагаємося з'ясувати, що саме клієнт прагне робити і якого типу система їм потрібна. У багатьох випадках ми намагаємося використовувати їхню мережу, але відокремлюємо її від нашої власної віртуальної локальної мережі, тому проблем у нас не так багато.

Після того, як ми визначимо, що шукають наші клієнти, беремо мережні керівництва, які вони надають, і розробляємо на їхній базі свій проект системи зв'язки з урахуванням усіх вимог. Можливо, мережа буде побудована на основі односпрямованої передачі даних. Можливо, ми розв'яжемо використовувати мультиміа. Можливо, це буде просте з'єднання точка-точка. А, можливо, наші системи для здійснення передачі даних будуть підтримувати різні способи пакетної маршрутизації. Іноді може знадобитися якийсь мобільний мережний набір, який наші клієнти зможуть брати із собою й піти в інший будинок, у такому випадку нам необхідно буде використовувати мобільний роутер або стільниковий маршрутизатор, який повністю захищений і може вертатися в систему. Усе залежить від того, що шукають наші клієнти.

Найкраще починати із сервісної служби, тому що ви відразу зіштовхнетеся з більшою кількістю проблем і несправностей. І надалі ви вже відразу зможете бачити, де проект реалізований правильно, а де щось може піти не так. Цей шлях утягне вас швидше, чим будь-який інший спосіб.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи мінімізації часу простою ІТ-інфраструктури, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		6

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

Огляд присвячений застосункам, що допомагають розв'язати різні завдання при роботі з мережними ПК, – одержати інформацію про віддалені комп'ютери, визначити обсяги трафіку, управляти підключеннями й моніторити дії користувачів.

Через те, що кожна з утиліт «заточена» для виконання різних функцій, замість звичної категорії «Функціонал» ми визначимо застосовність програм у різних умовах у блоці «Універсальність».

На початку добірки представлені інструменти для роботи із трафіком.

cFosSpeed

Переваги:

- автовизначення параметрів роутеру;
- робота з усіма розповсюдженими видами підключення.

Недоліки:

- якість виконання завдання прямо залежать від фізичних параметрів коннекта.

Призначення цієї утиліти – адміністрування мережних підключень усіх типів, а також оптимізація швидкості з'єднання. Застосунок застосовує спеціальний алгоритм поліпшення швидкості коннекта Traffic Shaping, який працює за принципом установки пріоритетів для переданих і отриманих пакетів.

Ще один позитивний ефект такої технології – зменшення часу відгуку віддаленого сервера, даний показник (ping) прямо впливає на комфорт при грі в онлайн-застосунку. Доступна функція перегляду завантаження каналу.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		7

Безкоштовно ознайомитися з функціями cFosSpeed можна протягом 30 днів. Вартість ліцензії – майже 18 доларів.

Throttle

Переваги:

- мінімум налаштувань.

Недоліки:

- утиліта найбільш ефективна при використанні застарілих типів з'єднання (Dial Up, ISDN);

- нелокалізоване меню.

Спеціалізований інструмент для підвищення швидкості коннекта. Застосунок працює з усіма розповсюдженими видами підключення й не залежить від характеристик використовуваного устаткування. Інтерфейс утиліти досить простий – користувачеві необхідно вказати тип з'єднання й вибрати свою версію операційної системи, тому відсутність української локалізації не можна вважати вагомим недоліком.

Trial-період використання програми – 28 днів. Вартість ліцензії – близько десяти доларів.

Тепер розглянемо софт для аналізу трафіку в online-режимі.

Wireshark

Переваги:

- потужний функціонал для онлайн-моніторингу.

- застосунок безкоштовний.

Недоліки:

- відсутність кирилиці.

- не занадто зручний інтерфейс для аналізу проблем на рівні застосунків.

Потужний мережний аналізатор, призначений для діагностики й захвата пакетів, оброблюваних роутером. Застосунок моніторить увесь трафік, що проходить між вузлами, містить інтегровану систему фільтрів і комплекс декодувальників протоколів, може шукати складності в роботі мережі й

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		8

«дзеркалювати» дані. Програма дозволяє імпортувати звіти, отримані за допомогою інших подібних утиліт.

Wireshark часто використовується системними адміністраторами для моніторингу стану мережі в онлайн-режимі. Утиліта є повністю безкоштовною.

CommView

Переваги:

- багатфункціональність;
- доповнення для аналізу VoIP-пакетів.

Недоліки:

- висока ціна ліцензії.

Скачати дану програму для моніторингу мережі можна безкоштовно, усі функції доступні без обмежень протягом місяця. Вартість повної версії становить 18000 гривень за стандартну версію й 36000 гривень за варіант із інтегрованим VoIP-аналізатором.

CommView for WiFi

Переваги:

- зручні інструменти для побудови діаграм;
- функція швидкого пошуку по звітах.

Недоліки:

- вартість;
- ні підтримки деяких адаптерів.

Спеціальна версія програми CommView, «заточена» на роботу з бездротовими точками доступу. Функціональність утиліти на високому рівні – сканування «ефіру», аналіз усього минаючого через хотспот трафіку, розкодувані пакетів, визначення завантаженості несучої, численні інструменти для візуалізації звітів і спеціальна функція визначення приналежності невідомих IP-адрес.

Дану програму для моніторингу мережі WiFi (з доступом в Інтернет) і діагностики трафіку до віддалених серверів можна скачати з інтерфейсом

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		9

українською мовою й безкоштовно використовувати протягом 30 днів. Ціна максимальної версії – 32000 гривень.

Прийшов час познайомитися із цікавою утилітою, яка допоможе заощадити засобу великим роботодавцям.

Activity Monitor

Переваги:

- різноманітні інструменти для стеження;
- можливість перегляду тексту відправлених листів.

Недоліки:

- висока вартість версій для великої кількості машин;
- утиліта працює тільки усередині локальної мережі.

Програма для перегляду дій інших користувачів на комп'ютерах у локальній мережі. Застосунок орієнтований на корпоративних клієнтів. Його часто використовує керівництво організацій для того, щоб довідатися, чим займаються співробітники за робочим місцем, а також якщо буде потреба в пошуку каналів витоку інформації. За допомогою Activity Monitor можна стежити за декількома десктопами одночасно, дивитися список запущених застосунків, моніторити месенджери й електронну пошту. Програма містить вбудований планувальник і інструменти для офлайн-аналізу.

Безкоштовно спробувати функціонал програми можна протягом тижня, ціна ліцензії залежить від кількості комп'ютерів і починається від 190 доларів.

Наостанку довідаємося особливості універсального комплексу для аналізу робочих місць і устаткування усередині «великих» мереж.

Spiceworks Network Monitor

Переваги:

- відмінна функціональність;
- безкоштовність.

Недоліки:

- недоцільно використовувати в малих мережах;

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		10

– керування проводиться через web-інтерфейс.

Багатофункціональний інструмент для обслуговування машин і устаткування. Програму можна використовувати для перегляду мережних адрес і імен комп'ютерів, аналізу актуальності версій софту на віддалених машинах (у тому числі й антивірусів) і добутку бекапів конфігурацій ПК. Застосунок підтримує функцію адміністрування системи MDM, може стежити за станом додаткового устаткування й інвентаризувати пристрою. Наприклад, перевіряти наявність чорнила в картриджі принтера. Сервіс містить вбудовану систему дистанційної допомоги співробітникам (Helpdesk).

Застосунок рекомендується використовувати в великих або середніх корпоративних мережах.

Через різну специфіку кожного інструмента наприкінці добірки ми відійдемо від традиційних рекомендацій певного додатка. Вибір оптимальної утиліти залежить від завдань, які необхідно розв'язати. Користувачі домашніх ПК можуть застосовувати програми для збільшення швидкості коннекта, інший софт призначений, насамперед, для роботи у великих мережах.

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.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		11

– Загальне призначення: можна використовувати для створення будь-яких програм.

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

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		12

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.

– 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.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		13

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 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,

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		14

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 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 на 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.

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 мінімізації часу простою ІТ-інфраструктури.

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-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		16

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

У процесі спостереження програми для моніторингу комп'ютерних мереж періодично виконують яка-небудь дія, наприклад, запитують певну сторінку на сервері або відправляють тестові повідомлення для перевірки поштового сервера компанії. Якщо система не одержує встановленого відповіді, вона реагує шляхом:

- відправлення сигналу тривоги відповідальному за роботу мережі ІТ фахівцеві;
- автоматичної активації системи захисту від збоїв і т.п.

Програми для моніторингу можна розглядати через область їх застосування – використовувані для малого бізнесу/домашніх мереж, розроблені для офісного середовища або звичайного виробництва й програми для спеціалізованих мереж – банківських, девелоперських і т.д. Окремо стоять багатофункціональні застосунки, які легко кастомізувати під будь-який тип бізнесу, включаючи й такий, який використовує ІоТ.

Доступні системному адміністраторові програми для моніторингу мережі можна умовно поділити на кілька груп, взявши за основу область їх застосування.

Перша група – популярні, лаконічні, відносно прості в освоєнні й не занадто багаті можливостями програми (Windows Network Monitor, Wireshark, Angry IP Scanner). Такі інструменти гарні в скромні по масштабах проектах – з їхньою допомогою зручно підтримувати домашню мережу, не відволікаючись на достаток зайвих (у цьому випадку) елементів інтерфейсу. У цій групі можна знайти багато вузькоспеціалізованих утиліт, призначених для конкретних цілей – виявлення загроз, аналіз трафіку, пентестінг і т.д.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		17

Також вони найчастіше не вимагають кастомізації й довгого налаштування, тому можуть бути використані для швидкого виявлення проблем у незнайомій мережі («на виїзді»).

Друга – програми, використовувані в основному для моніторингу великих мереж, що не мають специфічних особливостей (офіс, бізнес-центр і т.п.). У них більше уваги найчастіше приділене аналізу трафіку й безпеки, часто є можливість автоматично формувати «гарні» звіти й логи, а от можливість створювати й підключати плагіни може бути серйозно обмежена. Зате програма звичайно працює відразу «з коробки» і вже заточена під типову «офісну» мережу. До того ж при відсутності однієї-двох необхідних можливостей можна додатково встановити утиліти з першої групи – але їх роботу прийде відслідковувати й при необхідності включати у звіти/логи вручну. Окремо варто поцікавитися тем, як конкретний програмний застосунок справляється з «різношерстим» парком машин: у деяких випадках можна витратити зненацька багато часу на те, щоб linux-системи моніторились нарівні з іншими. Краще, щоб про це подбали розроблювачі.

Третя група призначена для моніторингу корпоративних мереж з якимись специфічними рисами: інтернет-провайдер, компанія, що надає послуги з оренди серверів, банк. Тут можна використовувати або програму, заздалегідь заточену під певний тип бізнесу (так, є кілька застосунків, націлених конкретно на розробку програмного забезпечення), або багатофункціональну програму, яка допускає доробку під потрібні цілі за допомогою плагінів (наприклад, PRTG Network Monitor). Багатофункціональні застосунки на зразок цих можна виділити в окрему групу: вони роблять ставку на достаток функцій і можливості, що впливають із нього – надійне зберігання всіх балок у єдиному виді, аналіз подій, швидке налаштування під незвичайні умови, наприклад, під «розумне виробництво» або високотехнологічний офіс, де більша частина вузлів є датчиками або складними автоматичними системами – і всі ці вузли потребують постійної комунікації один з одним. Тут варто звернути увагу на надійність

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		18

зберігання даних, здатність до роботи з більшими обсягами трафіку, віддалене керування (веб-інтерфейс або застосунки для планшета/смартфона) і готовність до кастомізації скриптами (або наявність декількох варіантів «з коробки»).

Природно, що кожний інструмент знаходить свого користувача, але ті інструменти, які відкриті до тим, хто ними користується, готові до змін і «говорять із користувачем».

3.2 Розробка структурної схеми

Базовий час безвідмовної роботи більше не є золотим стандартом. До 2026 року моніторинг мережі повинен робити більше, ніж просто звітувати про стан, він повинен пояснювати продуктивність у світі, що ґрунтується на гібридних технологіях. Мережі – це вже не просто статичні системи підтримки; це гнучкі, розподілені архітектури, які лежать в самому центрі взаємодії з клієнтами та бізнес-результатів.

Старі інструменти моніторингу більше не можуть встигати за всіма процесами. У 2026 році головне – не мати більше даних, а розуміти вже наявні дані. Мета полягає в тому, щоб поєднати точки та спростити керування мережею. Ось ключові тенденції, які визначатимуть, як ми керуватимемо, відстежуватимемо та спрощуватимемо мережевий стек у наступному році. Наведені нижче п'ять тенденцій представляють новий стандарт справності мережі, надаючи командам план переходу від реактивного усунення несправностей до проактивного, інтегрованого майбутнього.

- Тенденція 1: Від стану пристроїв до обізнаності про сервіс та враження
- Тенденція 2: Уніфікована видимість замінює розростання інструментів
- Тенденція 3: AIOPS стає основною оперативною здатністю
- Тенденція 4: Гібридне та хмарне підключення виходить на перший план
- Тенденція 5: Конфігурація, зміни та продуктивність збігаються

Давайте детально обговоримо, як проявляються ці тенденції.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		19

Тенденція 1: Від стану пристроїв до обізнаності про сервіс та враження

Роками моніторинг мережі зосереджувався на доступності пристроїв, використанні пропускну здатності та виявленні несправностей. Хоча ці показники все ще важливі, вони вже не відображають повної картини. Кінцеві користувачі скаржаться не на втрату пакетів чи помилки інтерфейсу, вони скаржаться на повільну роботу електронної пошти, затримки відеодзвінків або невідповідність бізнес-додатків. Щоб зрозуміти, звідки виникла проблема, потрібне єдине уявлення, яке співвідносить продуктивність мережі із затримкою додатків, забезпечуючи контекст, необхідний для визначення першопричини погіршення якості обслуговування в режимі реального часу.

У 2026 році моніторинг мережі все частіше вимірюється її здатністю відповісти на інше питання: як продуктивність мережі впливає на послуги та користувачів?

Сучасне середовище вимагає видимості, яка об'єднує:

- Затримка мережі та втрата пакетів.
- Продуктивність сервера та віртуальної машини.
- Час відповіді додатків.
- Індикатори взаємодії з користувачем.

Сучасний моніторинг переосмислює «здоров'я мережі», надаючи пріоритет сервісу та взаємодії з користувачем. Заглядаючи за межі окремих пристроїв, ці інструменти моніторингу забезпечують необхідну видимість для визначення того, що постраждало та чому, а не просто ідентифікують, в чому саме виникла проблема.

Тенденція 2: Уніфікована видимість усуває розкиданість інструментів

Більшість підприємств не планували створювати розгалужені стеки моніторингу, вони накопичуються з часом, наприклад, один інструмент для моніторингу мережі, інший для аналізу трафіку, ще один для серверів і ще один для журналів. Хоча кожен інструмент може добре працювати окремо, разом вони

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		20

створюють ізольовані системи, які уповільнюють усунення несправностей і призводять до поганої видимості.

Але до 2026 року організації активно переглядають цю модель. Замість додавання додаткових інструментів, компанії тепер переходять до кількох інтегрованих платформ, які пропонують єдину видимість, дозволяючи бачити все в одному місці.

Уніфікована видимість дозволяє:

- Швидший аналіз першопричин завдяки міждоменній кореляції.
- Менше панелей інструментів та передачі завдань між командами.
- Зменшений шум сповіщень.
- Менші операційні витрати та витрати на ліцензування.

Галузь усвідомлює, що ясність впливає з інтеграції, а не з простого обсягу.

Тенденція 3: AIOps стає основною оперативною здатністю

Штучний інтелект у моніторингу більше не є експериментальним. Попередні впровадження часто мали проблеми з довірою, прозорістю та практичною корисністю. У 2026 році очікування набагато чіткіші: Штучний інтелект (ШІ) повинен давати відчутні операційні результати.

Практичні випадки використання AIOps тепер включають:

- Зв'язування пов'язаних сповіщень у мережі, інфраструктурі та додатках.
- Зменшення надлишкових сповіщень під час каскадних збоїв.
- Зосередження уваги на основних причинах, а не лише на наявних симптомах.

– Вивчення нормальних моделей поведінки для швидкого виявлення аномалій.

Найголовніше, що роль AIOps полягає в усуненні операційної праці. Замість того, щоб інженери вручну запитували різні набори даних, щоб знайти першопричину, AIOps використовує машинне навчання для виявлення

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		21

закономірностей, на які можна посилити дії. Це переносить робоче навантаження команди з аналізу логів на вирішення інцидентів.

Тенденція 4: Гібридне та хмарне підключення виходить на перший план

Оскільки робочі навантаження розподіляються між гібридними та багатохмарними середовищами, мережа виступає критичним рівнем. Аналіз першопричин часто показує, що «проблеми з додатками» насправді є залежностями від мережевого шляху, де вузьким місцем є спосіб маршрутизації трафіку, а не сам код.

У 2026 році мережевий моніторинг повинен забезпечувати видимість:

- Продуктивність Інтернету та WAN.
- Використання прямого доступу до Інтернету (DIA).
- Хмарне з'єднання та затримка між регіонами.
- Моделі трафіку додатків.

Цей зсув підкреслює новий стандарт: ера ізольованого моніторингу продуктивності мережі та програм закінчилася. Щоб забезпечити справжню видимість, платформи повинні автоматично співвідносити локальну інфраструктуру із зовнішніми хмарними залежностями. Без цього зв'язку ваші дані – це просто шум; з ним вони стають практичним аналітичним матеріалом.

Тенденція 5: Конфігурація, зміни та продуктивність збігаються

Раніше ми використовували один інструмент для реєстрації змін конфігурації, а інший – для сповіщення про виникнення проблем. Такий поділ більше не працює. Об'єднуючи дані про зміни з показниками продуктивності, ми можемо перетворити «що змінилося» на негайну відповідь на питання «чому це сталося». Йдеться вже не про реагування на збої, а про розуміння впливу кожного розгортання.

Сучасне середовище вимагає розуміння:

- Дрейф конфігурації.
- Нещодавні зміни та їхній вплив.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		22

- Зниження продуктивності, пов'язане з певними оновленнями.

Очікується, що до 2026 року платформи мережевого моніторингу розумітимуть поведінку мережі, а не лише показники в режимі реального часу. Це дозволить командам виявляти ризиковані зміни раніше, пришвидшити аналіз першопричин та зменшити кількість повторних інцидентів.

Оскільки ці тенденції поєднуються, з'являється новий стандарт того, що повинна забезпечувати платформа моніторингу:

- Уніфікована видимість мережі, інфраструктури, програм та журналів.
- Аналіз трафіку, який пояснює, хто і що споживає пропускну здатність.
- Вбудований інтелектуальний підхід для кореляції сповіщень та виявлення аномалій.
- Обізнаність про стан конфігурації та історію змін.
- Можливість масштабування в гібридних та розподілених системах.

Те, що колись вважалося розширеними функціями, тепер є важливими основами для ефективної роботи ІТ-систем.

Питання, які ІТ-команди повинні поставити собі перед вибором стратегії моніторингу

Замість оцінювання інструментів лише на основі контрольних списків функцій, ІТ-командам слід ставити ширші питання:

- Чи пояснює ця платформа, чому виникають проблеми, чи сповіщає лише тоді, коли вони виникають?
- Чи може це співвіднести поведінку мережі з впливом на застосунки та користувачів?
- Чи зменшить це розповсюдження інструментів чи додасть ще один силос?
- Чи покращує вбудований інтелект результати, чи просто збільшує складність?

Відповіді на ці питання часто важливіші, ніж окремі показники чи інформаційні панелі.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		23

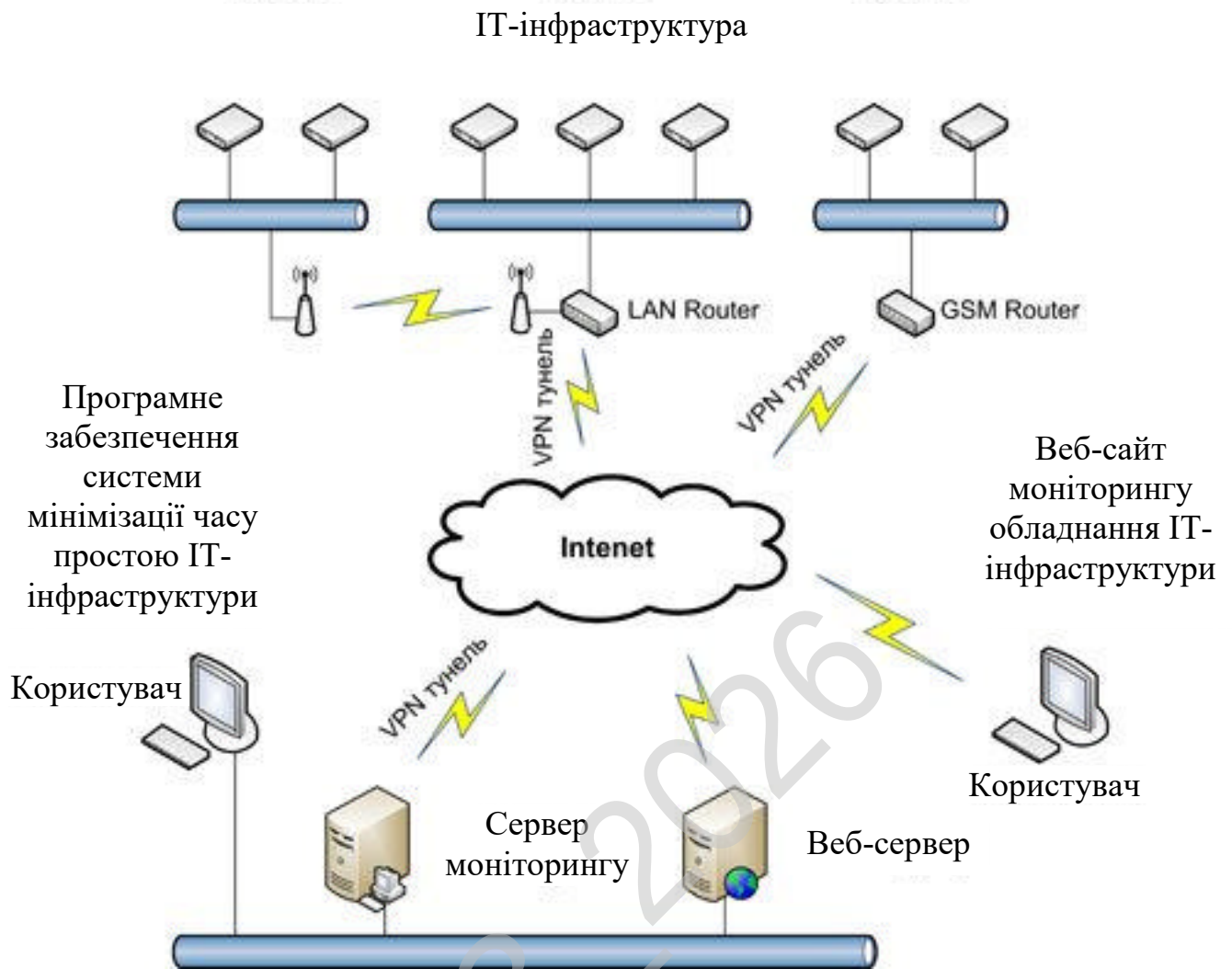


Рисунок 3.1 – Структурна схема системи

3.3 Розробка функціональної схеми

Система моніторингу мережі здійснює постійне спостереження за її роботою, щоб відразу ж виявити повільно функціонуючі або неробочі компоненти. Коли виявлена несправність, система сповіщає про неї адміністратора мережі за допомогою виконання заздалегідь установлених дій – наприклад, шляхом відправлення відповідного повідомлення на e-mail, месенджер, телефон, пейджер і ін. Також в автоматичному режимі активується система захисту від збоїв, яка тимчасово виводить проблемний елемент мережі з експлуатації доти, поки збій не ліквідують.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		24

Системи моніторингу мережі потрібні для пошуку проблем, що виникли в результаті перевантажень або відмови серверів устаткування, що супроводжує, і мережних з'єднань.

Дана система покликана вести спостереження й оперативно реагувати на загрози безпеки комп'ютерної мережі. ITInfrastructure відповідає за моніторинг пристроїв, задіяних у керуванні мережним захистом, а також хост-застосунків, що випускаються як компанією Cisco, так і іншими. Одним з основних переваг цього виду моніторингу є значне зниження обсягу неправильних спрацьовувань систем контролю безпеки за допомогою постійного наскрізного огляду локально-обчислювальної мережі.

Також дані системи моніторингу мережі дозволяють вчасно знаходити відхилення в роботі шляхом використання Network Behaviour Analysis. Система зарекомендувала себе як надійний інструмент застосування найбільш результативних способів ліквідації всіх виявлених і ймовірних загроз, що ґрунтуються на особливостях мережної конфігурації. ITInfrastructure забезпечує швидкий і простий доступ до утримуючої більш 150 готових форм, що налаштовуються звітів системі перевірці звітності.

До того ж система здатна давати коштовні рекомендації, що допомагають усувати проблеми за допомогою візуалізації найбільш імовірного способу мережної атаки й виявлення джерел загроз за допомогою спеціальних діаграм, що сприяє підвищенню ефективності роботи другого й наступних рівнів технічної підтримки.

Комплексна апаратна платформа ITInfrastructure дозволяє вести безперервне спостереження за використовуваною системою безпеки, стаючи ключовою ланкою її керування. Застосування подібних систем моніторингу мережі дозволяє IT підрозділу або, що займається технічним обслуговуванням підприємства зовнішньої компанії вчасно виявляти, управляти й знешкоджувати загрози безпеки його інформаційної системи.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		25

Якісний моніторинг повинен виявляти й ізолювати елементи, що заважають нормальній роботі обчислювальної мережі, даючи мережним адміністраторам корисні ради для їхньої повної ліквідації. Система відповідає політикам безпеки і як складовий компонент може бути включена в комплексну систему контролю над інформаційною інфраструктурою.

Проблеми, що часто зустрічаються в локальній мережі:

- недостатньо ефективні засоби виявлення мережних атак і збоїв;
- непродуктивність відповідних дій;
- складна в керуванні система безпеки;
- недотримання норм відповідності й відсутність порядку формування звітності;
- високий рівень складності й величезна швидкість поширення атак;
- серйозні наслідки збоїв;
- недолік кваліфікованих фахівців у сфері ІТ безпеки.

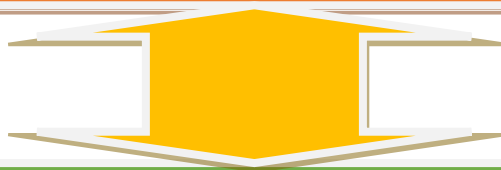
Використання систем моніторингу мережі дозволяє:

- інтегрувати у ЛОМ інтелектуальні функції для підвищення ефективності роботи механізму кореляції проблем і подій безпеки;
- відбивати атаки шляхом використання переваг мережної інфраструктури й діючої системи безпеки;
- одержувати візуальну представлення порушень безпеки й розслідувати їх в автоматичному режимі;
- вести спостереження над кінцевими вузлами, мережами й діями служби безпеки для відповідності їх установленим нормам.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		26

Блок визначення проблем, що часто зустрічаються
в мережі IT-інфраструктури :

- недостатньо ефективні засоби виявлення мережних атак і збоїв;
- непродуктивність відповідних дій;
- складна в керуванні система безпеки;
- недотримання норм відповідності й відсутність порядку формування звітності;
- високий рівень складності й величезна швидкість поширення атак;
- серйозні наслідки збоїв;
- недолік кваліфікованих фахівців у сфері IT безпеки.



Блок мінімізації часу простою IT-інфраструктури:

- інтегрування мережу інтелектуальні функції для підвищення ефективності роботи механізму кореляції проблем і подій безпеки;
- відбивання атаки шляхом використання переваг мережної інфраструктури й діючої системи безпеки;
- одержування візуального представлення порушень безпеки й розслідування їх в автоматичному режимі;
- ведення спостереження над кінцевими вузлами, мережами й діями служби безпеки для відповідності їх установленим нормам.

Рисунок 3.2 – Функціональна схема системи

Моніторинг мережі забезпечує перетворення, що відправляються ЛОМ і службою інформаційної безпеки даних у зручну для розуміння форму, що допомагає ефективно використовувати їх для усунення виявлених порушень, а також з метою забезпечення відповідності прийнятим нормам. Сучасні апаратні інструменти відбиття атак допомагають мережним адміністраторам вчасно виявляти загрози, визначати ступінь їх пріоритетності й швидко ліквідувати їх за допомогою спеціальних дій.

3.4 Розробка діаграми процесів

Відповідно до методичних рекомендацій розроблення графічної частини кваліфікаційної бакалаврської роботи розглянемо розроблену діаграму процесів яка зображена на рисунку 3.3.

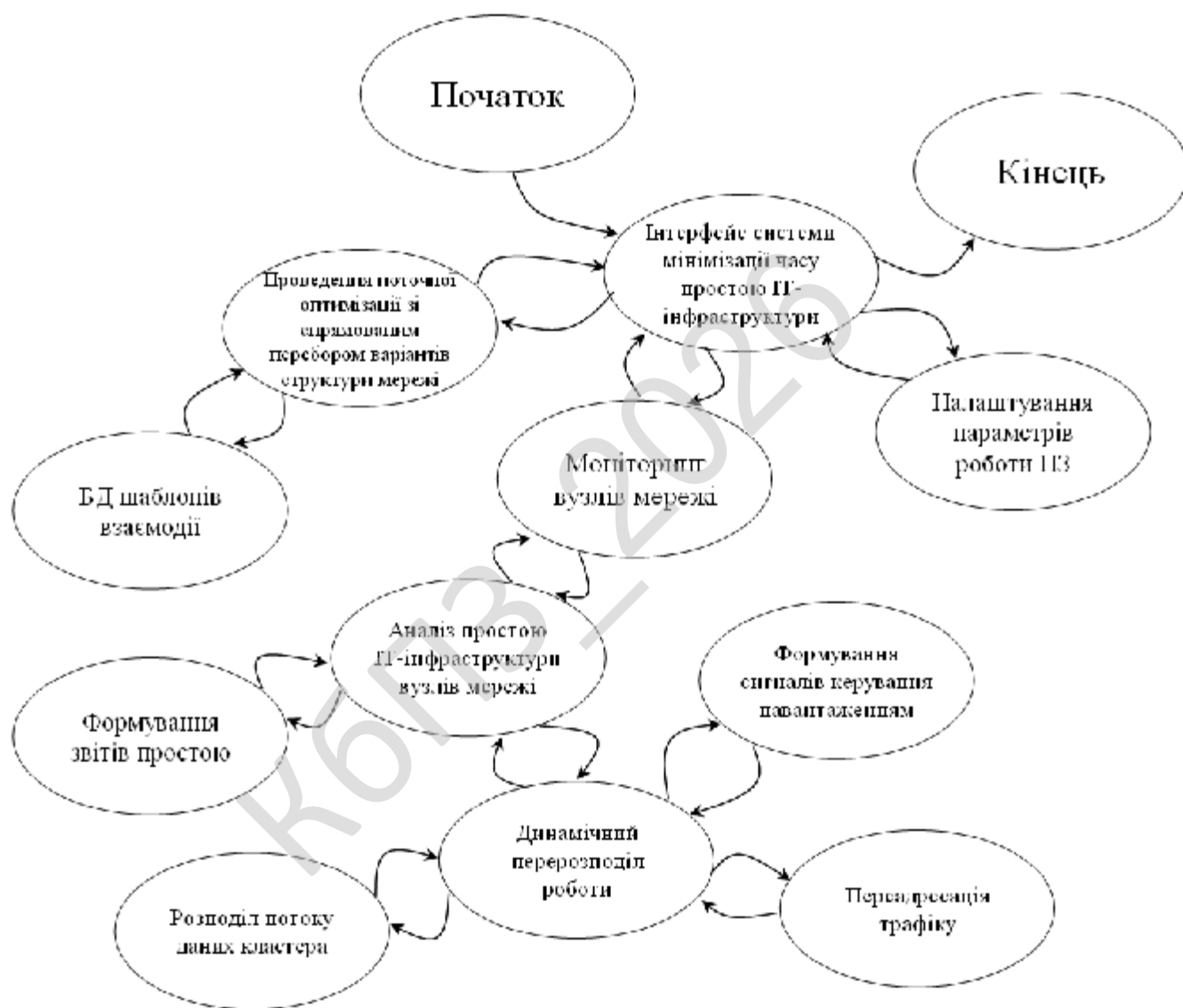


Рисунок 3.3 – Діаграма взаємодії процесів

Розроблена діаграма взаємодії процесів використовується для представлення та візуалізації процесів обробки даних тобто структурного проектування бакалаврської роботи.

Основні складові елементи діаграми взаємодії процесів це потоки даних:

- Репозиторії, потік сховища даних.
- Потоки зовнішні по відношенню до системи сутності.
- Процеси які являють собою трансформацію даних в рамках описуваної системи.
- Потоки даних гібридні між елементами трьох попередніх типів.

Відповідно до документації основна будова діаграми процесів полягає у графічному представленні складу сукупностей даних, що характеризуються як співвідношення різних частин кожної з сукупностей. Склад статистичної сукупності графічно може бути представлений як за допомогою абсолютних, так і відносних показників. Графічне зображення складу сукупності по абсолютними і відносними показниками сприяє проведенню більш глибокого аналізу і дозволяє проводити аналіз системи.

Для схематичного представлення системи що розробляється необхідно спочатку представити діаграму взаємодії процесів даних рівня контексту, завдяки чому буде показано взаємодію системи в цілому у подальшому. Використовується модель проектування, графічне представлення «потоків» даних в інформаційній системі. Розроблена діаграма взаємодії процесів системи в подальшому уточнюється шляхом деталізації процесів та потоків даних з метою показати систему що розробляється. Таким чином у результаті після розгляду, вищеописаної системи, схеми структурної, функціональної, діаграми взаємодії процесів перейдемо до опису та розгляду блок-схем основної програми, та підпрограм, які використовуються, для реалізації системи.

4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

Розглянемо реалізацію бакалаврської дипломної роботи. Були проведені розрахунки і підібрані набори тестових даних для перевірки правильності реалізації проектних рішень.

Було створено блок-схеми роботи системи. 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.

Блок-схеми показують весь процес роботи системи з підсистемами та частково доказують правильність вибраних проектних рішень. 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 високого рівня декомпозиції задач на класи.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		30

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

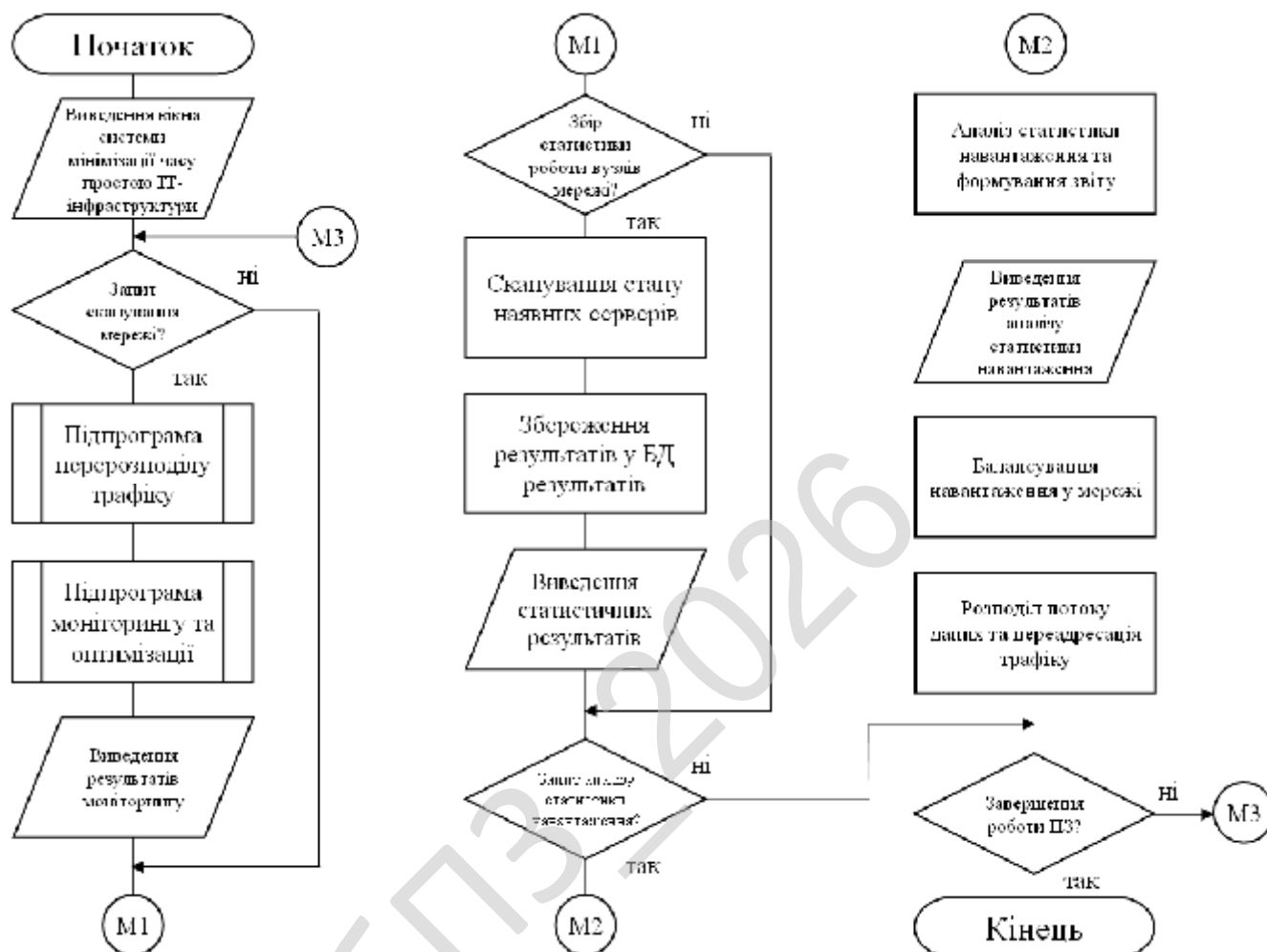


Рисунок 4.1 – Блок-схема основної програми

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 підсистеми 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.

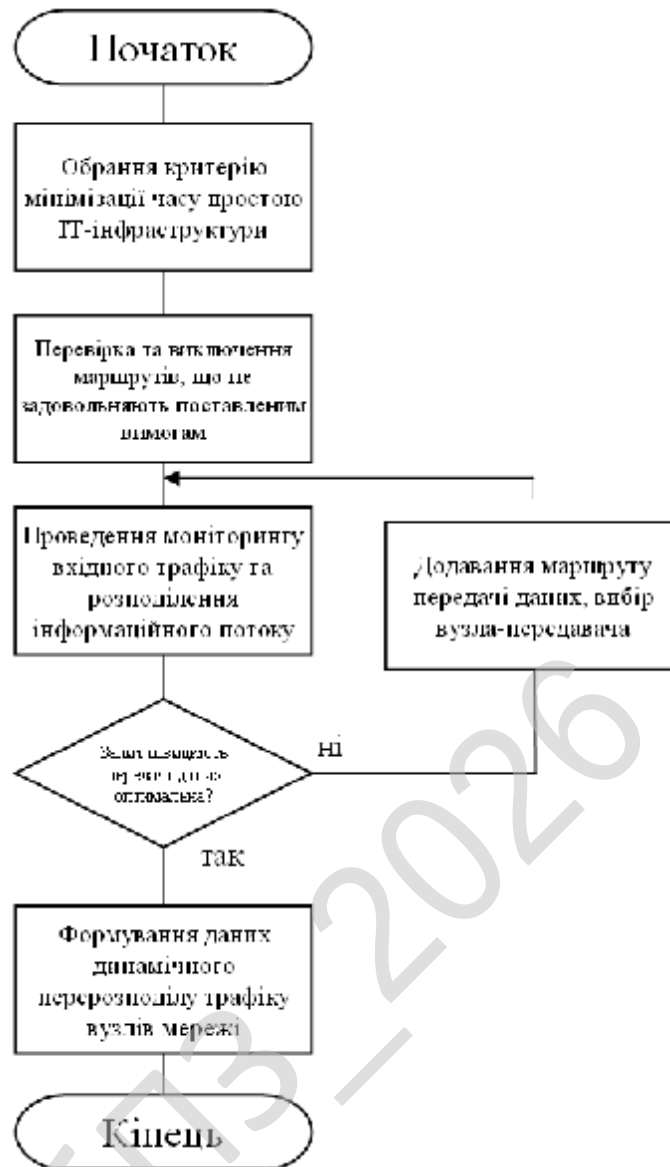


Рисунок 4.2 – Блок-схема роботи підпрограми

Архітектура клієнт-сервер є одним із архітектурних шаблонів програмного забезпечення та є домінуючою концепцією у створенні розподілених мережних програм і передбачає взаємодію та обмін даними між ними. Вона передбачає такі основні компоненти:

- набір серверів, які надають інформацію або інші послуги програмам, які звертаються до них;
- набір клієнтів, які використовують сервіси, що надаються серверами;
- мережа, яка забезпечує взаємодію між клієнтами та серверами.

Сервери є незалежними один від одного. Клієнти також функціонують паралельно і незалежно один від одного. Немає жорсткої прив'язки клієнтів до серверів. Більш ніж типовою є ситуація, коли один сервер одночасно обробляє запити від різних клієнтів; з іншого боку, клієнт може звертатися то до одного сервера, то до іншого. Клієнти мають знати про доступні сервери, але можуть не мати жодного уявлення про існування інших клієнтів.

Дуже важливо ясно уявляти, хто або що розглядається як «клієнт». Можна говорити про клієнтський комп'ютер, з якого відбувається звернення до інших комп'ютерів. Можна говорити про клієнтське та серверне програмне забезпечення. Нарешті, можна говорити про людей, які бажають за допомогою відповідного програмного та апаратного забезпечення отримати доступ до тієї чи іншої інформації.

Загальноприйнятим є положення, що клієнти та сервери – це перш за все програмні модулі. Найчастіше вони знаходяться на різних комп'ютерах, але бувають ситуації, коли обидві програми – і клієнтська, і серверна, фізично розміщуються на одній машині; в такій ситуації сервер часто називається локальним.

Модель клієнт-серверної взаємодії визначається перш за все розподілом обов'язків між клієнтом та сервером. Логічно можна відокремити три рівні операцій:

- рівень представлення даних, який по суті являє собою інтерфейс користувача і відповідає за представлення даних користувачеві і введення від нього керуючих команд;
- прикладний рівень, який реалізує основну логіку ПЗ і на якому здійснюється необхідна обробка інформації;
- рівень управління даними, який забезпечує зберігання даних та доступ до них.

Дворівнева клієнт-серверна архітектура передбачає взаємодію двох програмних модулів – клієнтського та серверного. В залежності від того, як між ними розподіляються наведені вище функції, розрізняють:

- модель тонкого клієнта, в рамках якої вся логіка ПЗ та управління даними зосереджена на сервері. Клієнтська програма забезпечує тільки функції рівня представлення;

- модель товстого клієнта, в якій сервер тільки керує даними, а обробка інформації та інтерфейс користувача зосереджені на стороні клієнта. Товстими клієнтами часто також називають пристрої з обмеженою потужністю: кишенькові комп'ютери, мобільні телефони та ін.

Типовим прикладом клієнт-серверної взаємодії є WWW. Існує величезна кількість веб-серверів, на яких розміщується та чи інша інформація. У найпростішому випадку ця інформація являє собою набір веб-сторінок, які можуть зберігатися на сервері у вигляді файлів, розмічених за допомогою мови розмітки HTML. Але ситуація, як правило, є складнішою; значна частина веб-ресурсів на сучасному етапі є динамічними, тобто вони не існують в заздалегідь підготовленому вигляді, а створюються безпосередньо в процесі обробки запиту від користувача.

Для того, щоб людина, яка працює в Інтернеті, могла переглянути ту чи іншу сторінку, на її комп'ютері повинно бути встановлено відповідне програмне забезпечення. Програми для перегляду веб-сторінок називаються браузером.

Але, крім браузерів, до серверів можуть звертатися і інші клієнти, а саме – автономні програми. Вони можуть передбачати взаємодію з людиною, а можуть працювати в цілком автоматичному режимі. Типовим класом таких програм є роботи, призначені для автоматичного перегляду веб-ресурсів. Зокрема, роботи є важливим елементом пошукових систем і використовуються ними для перегляду сторінок і збору інформації про них.

Для запиту до веб-сервера клієнтська програма повинна задати місцезнаходження комп'ютера, на якому розміщується серверна програма, назву

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		34

потрібного документа і, можливо, інші дані, які специфікують запит. Мережа забезпечує знаходження сервера і передачу йому клієнтського запиту. Серверні програми обробляють цей запит, відповідь пересилається по мережі клієнтові.

Трирівнева клієнт-серверна архітектура, яка почала розвиватися з середини 90-х років, передбачає відділення прикладного рівня від управління даними. Відокремлюється окремий програмний рівень, на якому зосереджується прикладна логіка ПЗ. Програми проміжного рівня можуть функціонувати під управлінням спеціальних серверів ПЗ, але запуск таких програм може здійснюватися і під управлінням звичайного веб-сервера. Нарешті, управління даними здійснюється сервером даних.

Для роботи з системою користувач використовує стандартне програмне забезпечення –звичайний браузер. Це позбавляє його необхідності завантажувати та інсталювати спеціальні програми (хоча інколи така необхідність все-таки виникає).

Але користувачеві слід надати в розпорядженні інтерфейс, який дозволяв би йому взаємодіяти з системою і формувати запити до неї. Форми, що визначають цей інтерфейс, розміщуються на веб-сторінках та завантажуються разом з ними.

Веб-оглядач формує запит та пересилає його до сервера, який здійснює обробку. При необхідності сервер викликає серверні програмні модулі, які забезпечують обробку запиту і в разі потреби звертаються до сервера даних. Сервер даних здійснює операції з даними, що зберігаються в системі та складають її інформаційну основу. Зокрема, він може здійснити вибірку з інформаційної бази відповідно до запиту та передати її модулю проміжного рівня для подальшої обробки. Дані, з якими працює сервер даних, найчастіше організовані як реляційна база даних.

Найчастіше веб-сервер і серверні модулі проміжного рівня розміщуються на одному комп'ютері, хоч і являють собою окремі і логічно незалежні програмні модулі.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		35

На сучасному етапі для програмування модулів проміжного рівня використовується мова серверних сценаріїв PHP, а для управління даними – СУБД MySQL. Таким чином, зв'язку PHP-MySQL слід розглядати як стандартний інструмент для створення порівняно простих інтерактивних веб-сайтів та систем електронної комерції; близько 90% комерційних систем сьогодні створюється саме на цій основі. Водночас як засоби управління даними, так і middleware-засоби можуть бути найрізноманітнішими. Так, для створення серверних програм, крім PHP, широко застосовуються Java, Perl, Python, Delphi.

Взагалі, технології створення розподілених, зокрема веб-програм, стрімко розвиваються. Слід згадати про технології EJB (Enterprise Java Beans), CORBA, а також про .NET – порівняно нову ініціативу компанії Microsoft. Для зберігання даних та їх передачі часто використовується так звана розширювана мова розмітки XML (Extensible Markup Language).

Розглянемо визначення API. Це прикладний програмний інтерфейс (Application Programming Interface, API) – набір визначень підпрограм, протоколів взаємодії та засобів для створення програмного забезпечення.

Спрощено – це набір чітко визначених методів для взаємодії різних компонентів. API надає розробнику засоби для швидкої розробки програмного забезпечення. API може бути для веб-базованих систем, операційних систем, баз даних, апаратного забезпечення, програмних бібліотек.

Одним з найпоширеніших призначень API є надання набору широко використовуваних функцій, наприклад для малювання вікна чи іконок на екрані. API є абстрактним поняттям — програмне забезпечення, що пропонує деякий API, часто називають реалізацією (implementation) даного API. У багатьох випадках API є частиною набору розробки програмного забезпечення, водночас, набір розробки може включати як API, так і інші інструменти/апаратне забезпечення, отже ці два терміни не є взаємозамінювані.

Високорівневі API часто програють у гнучкості. Виконання деяких функцій нижчого рівня стає набагато складнішим, або навіть неможливим.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		36

В об'єктно-орієнтованих мовах, прикладний програмний інтерфейс зазвичай включає в себе опис набору визначень класу, з набором форм поведінки, пов'язаних з цими класами. Це абстрактне поняття пов'язане з реальними функціями, які надані або надаватимуться, класами, які реалізуються в методах класу.

Прикладний програмний інтерфейс в даному випадку можна розглядати як сукупність всіх методів, які публічно доступні в класах (зазвичай званий інтерфейс класу). Це означає, що прикладний програмний інтерфейс вказує методи, за допомогою яких взаємодіє з об'єктами, отриманими з визначень класів і обробляє їх.

У більш загальному плані можна визначити Прикладний Програмний Інтерфейс як сукупність усіх видів об'єктів, які можна вивести з визначення класу, і пов'язаних з ними можливих варіантів поведінки.

Наприклад: клас, що представляє Stack, може просто виставити публічно два методи Push() (для додавання нового елемента в стек) і Pop() (для вилучення останнього пункту, ідеально розташований на вершині стека).

У цьому випадку Прикладний Програмний Інтерфейс може бути інтерпретованим як два методи pop() і push(), або, більш широко, використовується варіант, коли можна використовувати елемент типу Stack, який реалізує поведінку стека, надаючи йому можливість для додавання / видалення елементів з вершини. Друга інтерпретація видається більш доречною в дусі об'єктно-орієнтованого підходу.

Якість документації, пов'язаної з Прикладним Програмним Інтерфейсом, є часто ключовим фактором, що визначає його успішність з точки зору простоти використання.

ППІ, як правило, пов'язаний із бібліотеками програмного забезпечення: ППІ описує і вказує очікувану поведінку в той час, як бібліотека є фактичною реалізацією даного набору правил. Один ППІ може мати декілька реалізацій (або

жодної, будучи абстрактним) у вигляді різних бібліотек, які мають такий же інтерфейс.

Прикладний програмний інтерфейс також може бути пов'язаним з платформами програмування: платформа може бути заснована на кількох бібліотеках реалізує декілька інтерфейсів ППІ, але на відміну від звичайного використання ППІ, доступ до поведінки вбудований в платформу опосередкований шляхом розширення його змісту новими класами і вставлений в саму платформу. Крім того, загальний потік управління програми може бути під контролем абонента.

Прикладний програмний інтерфейс може бути також реалізацією протоколу.

Коли ППІ реалізує протокол, він може бути заснованим на проксі-методах віддалених викликів, що засновані на протоколі зв'язку. Роль ППІ може заключатися саме в тому, щоб приховати деталі транспортного протоколу. Наприклад: RMI є ППІ, який реалізує протокол або JRMP IIOP як RMI-IIOP.

Протоколи, як правило, розподіляються між різними технологіями і зазвичай дозволяють різним технологіям обмінюватися інформацією, діючи як абстракція між двома світами. ППІ, як правило, є специфічним для конкретної технології: звідси, інтерфейси даної мови не можуть бути використані на інших мовах, якщо виклики функції не будуть перетворені з конкретної адаптації бібліотеки.

Деякі мови, серед яких такі, що працюють на віртуальних машинах (наприклад: мови, сумісні з NET CLI середовища CLR і JVM сумісних мов у віртуальній машині Java) можуть ділитися програмними інтерфейсами.

У цьому випадку віртуальна машина дозволяє мові взаємодії завдяки спільному знаменнику віртуальної машини, що абстрагується від конкретної мови, використовувати проміжний байт-код і його мову.

При використанні прикладного програмного інтерфейсу в контексті веб-розробки, як правило, ППІ визначається набором повідомлень запиту HTTP,

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		38

також визначається структура повідомлень-відповідей, зазвичай у розширенні мови розмітки XML або в форматі об'єктного запису JavaScript (JSON). У той час як прикладний програмний інтерфейс у Web історично був практично синонімом для веб-служби, останнім часом тенденція змінилась (так званий Web 2.0) на відхід від Simple Object Access Protocol (SOAP) на основі веб-сервісів і сервіс-орієнтованої архітектури (SOA) на більш прямі передачі репрезентативного стану (REST) стилів веб-ресурсів та ресурсів-орієнтованої архітектури (ROA).

Частина цієї тенденції пов'язана з рухом Семантичного веб-ресурсу до Опису Платформ (RDF), Концепції розвитку веб-технологій інженерних онтологій. Прикладні програмні інтерфейси у Web, що дозволяють комбінувати декількома прикладними програмними інтерфейсами в нові додатки називають гібридними.

Незважаючи на те що я працював над ПЗ один в реалізації програми я використовував підходи пришвидшення розробки на основі методологій Agile.

Гнучка розробка програмного забезпечення (Agile software development, agile-методи) – клас методологій розробки програмного забезпечення, що базується на ітеративній розробці, в якій вимоги та розв'язки еволюціонують через співпрацю між самоорганізовуваними багатофункціональними командами.

Гнучка розробка – найкращий засіб для підвищення продуктивності розробників програмного забезпечення.

Більшість гнучких методологій націлені на мінімізацію ризиків, шляхом зведення розробки до серії коротких циклів, що мають назву ітерацій, які зазвичай тривають один-два тижні. Кожна ітерація сама по собі виглядає як програмний проект в мініатюрі, і включає всі завдання, необхідні для видачі мінімального приросту за функціональністю: планування, аналіз вимог, проектування, кодування, тестування і документування. Хоча окрема ітерація, як правило, недостатня для випуску нової версії продукту, мається на увазі те, що гнучкий програмний проект готовий до випуску наприкінці кожної ітерації. Після закінчення кожної ітерації, команда виконує переоцінку пріоритетів розробки.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		39

Agile акцентує увагу на безпосередньому спілкуванні «віч-на-віч». Більшість agile команд розташовані в одному офісі, його іноді називають bullpen. Як мінімум вона включає і «замовників» (замовники, які визначають продукт, також це можуть бути менеджери продукту, бізнес аналітики або клієнти). Офіс може також включати тестувальників, дизайнерів інтерфейсу, технічних авторів і менеджерів.

Основною метрикою agile методів є робочий продукт. Віддаючи перевагу безпосередньому спілкуванню, agile-методи зменшують обсяг письмової документації в порівнянні з іншими методами. Це привело до критики цих методів як недисциплінованих.

Agile – родина процесів розробки, а не єдиний підхід в розробці програмного забезпечення, і визначається Agile Manifesto. Agile не включає практик, а визначає цінності та принципи, якими керуються успішні команди.

Agile Manifesto розроблений і прийнятий 17 розробниками 11-13 лютого 2001 року на лижному курорті The Lodge at Snowbird в горах Юти. Маніфест підписали представники наступних методологій Extreme programming, Scrum, DSDM, Adaptive software development, Crystal Clear, Feature driven development, Pragmatic Programming. Agile Manifesto містить 4 основні ідеї та 12 принципів. Примітно, що Agile Manifesto не містить практичних порад.

Основні ідеї:

- Особистості та їхні взаємодії важливіші, ніж процеси та інструменти;
- Робоче програмне забезпечення важливіше, ніж повна документація;
- Співпраця із замовником важливіша, ніж контрактні зобов'язання;
- Реакція на зміни важливіша, ніж дотримання плану.

Принципи, які роз'яснює Agile Manifesto:

- задоволення клієнта за рахунок ранньої та безперебійної поставки коштовного програмного забезпечення;
- вітання змін вимог навіть наприкінці розробки (це може підвищити конкурентоспроможність отриманого продукту);

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		40

- часта поставка робочого програмного забезпечення (кожен місяць або тиждень або ще частіше);
- тісне, щоденне спілкування замовника з розробниками впродовж всього проекту;
- проектом займаються мотивовані особистості, які забезпечені потрібними умовами роботи, підтримкою і довірою;
- рекомендований метод передачі інформації – особиста розмова (віч-на-віч);
- робоче програмне забезпечення – найкращий вимірювач прогресу;
- спонсори, розробники та користувачі повинні мати можливість підтримувати постійний темп на невизначений термін;
- постійну увагу поліпшенню технічної майстерності та зручному дизайну;
- простота – мистецтво не робити зайвої роботи;
- найкращі технічні вимоги, дизайн та архітектура виходять у самоорганізованої команди;
- постійна адаптація до мінливих обставин.

Маніфест та Принципи гнучкої розробки містять високорівневі ідеї щодо того, як потрібно вибудовувати процес розробки програмного забезпечення, щоб успішно завершувати проекти й створювати команди, в яких приємно та цікаво працювати.

Документи визначають, що потрібно для цього зробити, але не говорять, як це зробити. По-іншому й не могло бути, оскільки Маніфест та Принципи народилися внаслідок консенсусу представників різних (хоча й споріднених) напрямів, які могли знайти спільну основу лише на рівні базових цінностей та принципів.

Критика. Багато керівників проектів, що працюють у традиційних методологіях на кшталт «водоспаду», критикують agile-методи.

Один з повторюваних пунктів критики: при agile-підході часто нехтують створенням «дорожньої карти» розвитку продукту, так само як і управлінням

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		41

вимогами, в процесі якого і формується така «карта». Гнучкий підхід до управління вимогами не має на увазі далекосяжних планів (по суті, управління вимогами просто не існує в даній методології), а має на увазі можливість замовника раптом і несподівано наприкінці кожної ітерації виставляти нові вимоги, що часто суперечать архітектурі вже створеного і поставленого продукту. Таке іноді призводить до катастрофічних «авралів» з масовим рефакторингом і переробками практично на кожній черговій ітерації.

Крім того вважається, що робота в agile мотивує розробників вирішувати всі прибулі завдання найпростішим і найшвидшим можливим способом, при цьому часто не звертаючи уваги на коректність коду з точки зору вимог базової платформи (підхід «працює, та й добре», при цьому не враховується, що може перестати працювати при найменшій зміні або ж породити важкі до відтворення дефекти після реального розгортання у клієнта). Це призводить до зниження якості продукту і накопиченню дефектів.

Методології. Існують методології, які дотримуються цінностей і принципів заявлених в Agile Manifesto, деякі з них:

1. Agile Modeling – набір понять, принципів і прийомів (практик), що дозволяють швидко і просто виконувати моделювання і документування в проектах розробки програмного забезпечення. Не включає в себе детальну інструкцію з проектування, не містить описів, як будувати діаграми на UML.

Основна мета – ефективне моделювання і документування; але не охоплює програмування та тестування, не включає питання управління проектом, розгортання і супроводу системи. Однак включає в себе перевірку моделі кодом.

2. Agile Unified Process (AUP) спрощена версія IBM Rational Unified Process (RUP), розроблена Скоттом Амблером, яка описує просте і зрозуміле наближення (модель) для створення програмного забезпечення для бізнес-додатків.

3 Agile Data Method – група ітеративних методів розробки програмного забезпечення, в яких вимоги та рішення досягаються в рамках співпраці різних крос-функціональних команд.

4. DSDM заснований на концепції швидкої розробки додатків (Rapid Application Development, RAD). Являє собою ітеративний і інкрементний підхід, який надає особливого значення тривалій участі в процесі користувача/споживача.

5. Essential Unified Process (EssUP).

6. Екстремальне програмування (Extreme programming, XP).

7. Feature driven development (FDD) – функціонально-орієнтована розробка. Використовуване в FDD поняття функції або властивості (feature) Системи досить близько до поняття прецеденту використання, використовуваному в RUP, істотна відмінність – це додаткове обмеження: «кожна функція повинна допускати реалізацію не більше, ніж за два тижні». Тобто якщо сценарій використання досить малий, його можна вважати функцією. Якщо ж великий, то його треба розбити на декілька відносно незалежних функцій.

8. Getting Real – ітераційний підхід без функціональних специфікацій, що використовується для веб-додатків. У даному методі спершу розробляється інтерфейс програми, а потім її функціональна частина.

9. OpenUP – це ітераційно-інкрементний метод розробки програмного забезпечення. Позиціюється, як легкий і гнучкий варіант RUP. OpenUP ділить життєвий цикл проекту на чотири фази: початкова фаза, фази уточнення, конструювання та передачі. Життєвий цикл проекту забезпечує надання зацікавленим особам та членам колективу точок ознайомлення і прийняття рішень впродовж усього проекту. Це дозволяє ефективно контролювати ситуацію і вчасно приймати рішення про задовільність результатів. План проекту визначає життєвий цикл, а кінцевим результатом є остаточний додаток.

10. Scrum встановлює правила керування процесом розробки та дозволяє використовувати вже існуючі практики кодування, коректуючи вимоги або

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		43

вносячи тактичні зміни. Використання цієї методології дає можливість виявляти і усувати відхилення від бажаного результату на більш ранніх етапах розробки програмного продукту.

11. Бережлива розробка програмного забезпечення (lean software development). Використовує підходи з концепції бережливого виробництва.

Подійно-орієнтована архітектура (Event-driven architecture, надалі EDA) – шаблон архітектури програмного забезпечення, який призначений для створення подій, їх виявлення, споживання і реагування на них.

Подія може бути визначена як значна зміна стану. Наприклад, коли споживач купує автомобіль, стан автомобіля змінюється з "на продаж" до "продано". Архітектура системи дилера автомобілів може трактувати цю зміну стану як подію, поява якої може стати відомою іншим програмам даної архітектури.

З формальної точки зору, те, що виробляється, публікується, поширюється, виявляється і споживається (як правило, асинхронно) є повідомленням, яке називають сповіщенням про подію (або нотифікацією), а не самою подією, яка є зміною стану, що викликає появу повідомлення.

Події не подорожують, вони просто відбуваються. Проте термін подія часто використовується метонімічно для позначення самого нотифікаційного повідомлення, що може призвести до певної плутанини.

Цей архітектурний шаблон може застосовуватися при проектуванні і реалізації ПЗ і систем, які передають події між слабкозв'язаними компонентами програмного забезпечення і сервісами (службами).

Подійно-орієнтована система як правило складається з емітерів подій (або агентів) і споживачів подій (або стоків).

Стоки несуть відповідальність за здійснення реагування на появу події. Реакція не завжди може бути повністю забезпечена самим стоком. Наприклад, стік, може бути відповідальним лише за фільтрацію, трансформацію і відправку події до іншого компонента або він може забезпечити повністю самостійну

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		44

реакцію на таку подію. Перша категорія стоків може бути заснована на традиційних компонентах, таких як проміжне програмне забезпечення, орієнтоване на обробку повідомлень (message oriented middleware, MOM), в той час, як друга категорія стоків (самостійна реакція в режимі он-лайн) може вимагати більш придатної платформи (фреймворку) для виконання транзакцій.

Розробка ПЗ і систем в подійно-орієнтованій архітектурі дозволяє їм бути сконструйованими способом, який більш відповідає вимогам до їх створення, оскільки такі системи в більшій мірі пристосовуються до непередбачуваних і асинхронних середовищ.

Подійно-орієнтована архітектура (EDA) може доповнювати сервісно-орієнтовану архітектуру (SOA), оскільки сервіси (служби) можуть бути активовані тригерами, які ініціюються при настанні подій.

Ця парадигма особливо корисна, коли стік не забезпечує власного виконання будь-яких дій.

Подійно-орієнтована SOA (SOA-2) розвиває архітектури SOA і EDA для забезпечення більш глибокого і надійного рівня сервісів за рахунок використання раніше невідомих причинно-наслідкових зв'язків, щоб сформувати новий шаблон подій. Цей новий шаблон бізнес-аналітики дає поштовх до небаченого раніше зростання рівня автоматизації підприємства за рахунок привнесення додаткової цінної інформації в описану раніше модель діяльності.

Обчислювальна техніка та сенсорні пристрої (сенсори, датчики, контролери) можуть виявляти зміни стану об'єктів або умов і створювати події, які потім можуть бути оброблені сервісом (службою) або системою.

Подія може складатися з двох частин: заголовка події та тіла події. Заголовок події може включати в себе інформацію таку як, наприклад, назва події, часова мітка події і тип події. Тіло події – це частина, яка описує факт, що стався в дійсності. Тіло події не слід плутати з шаблоном або логікою, яка може бути застосована як реакція на саму подію.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		45

Архітектура, керована подіями, складається з чотирьох логічних рівнів (шарів). Вона починається з виявлення факту, його технічного подання у формі події і закінчується не пустою множиною реакцій на цю подію.

Генератор подій.

Першим логічним шаром є генератор подій, який виявляє факт і представляє цей факт подією. Оскільки фактом може бути практично все, що може бути сприйнято, то ним може бути і генератор подій. Наприклад, генератором може бути клієнт електронної пошти, система електронної комерції або певний тип датчика.

Перетворення різних даних, отриманих від датчиків, в єдину стандартизовану форму даних, які можуть бути оцінені, є основною проблемою при розробці та реалізації цього шару. Однак, враховуючи, що подія є строго декларативною, можна легко застосовувати будь-які операції трансформації, тим самим усуваючи необхідність забезпечення високого рівня стандартизації.

Канал подій.

Канал подій – це механізм, через який інформація від генератора подій передається до обробника подій (event engine) або стоку.

Це може бути з'єднання TCP/IP або вхідний файл будь-якого типу (простий текст, формат XML, e-mail тощо). В один і той же час може бути відкрито кілька каналів подій. Як правило, оскільки обробник подій повинен працювати в режимі, наближеному до реального часу, канали подій зчитуються асинхронно. Події зберігаються в черзі, очікуючи наступної обробки механізмом обробки подій.

Механізм обробки подій.

Механізм обробки подій (event processing engine) є місцем, де подія ідентифікується і вибирається відповідна реакція на нього, яка потім виконується. Це також може призвести до породження ряду тверджень. Якщо подія, яка надійшла до механізму обробки подій, є наприклад такою «Запаси продукту ID досягли нижнього допустимого рівня», це може ініціювати, наприклад, такі реакції як «Замовити продукт ID» і «Сповістити персонал».

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		46

Наступна подійно-орієнтована дія (післядія).

Щодо того, як можуть проявлятися наслідки події, слід відмітити, що вони можуть проявитись багатьма різними способами і у різноманітних формах (наприклад, повідомлення електронної пошти, надіслане комусь, або ПЗ, що виводить деяке попередження на екран). Залежно від рівня автоматизації, який забезпечується стоком (механізмом обробки подій), ці дії можуть виявитись зайвими.

Є три основні стилі обробки подій: простий, потоковий і складний. Часто ці три стилі використовуються спільно у розвинутій подійно-орієнтованій архітектурі.

Проста обробка подій.

Проста обробка подій стосується подій, які безпосередньо належать до специфічних вимірних змін умов. У випадку простої обробки подій, мають справу з появою відомих подій, що ініціюють післядію (післядії). Проста обробка подій зазвичай використовується для управління потоком робіт в реальному часі, скорочуючи тим самим час затримки і вартість робіт.

Наприклад, прості події можуть створюватись (породжуватись) датчиком, що виявляє зміну тиску в шині або температуру навколишнього середовища.

Обробка потоку подій.

При обробці потоку подій (event stream processing, далі ESP) відбуваються як звичайні, так і відомі події. Звичайні події (заявки, передачі RFID) перевіряються на те, чи є вони відомими, і передаються інформаційним передплатникам. Обробка потоку подій зазвичай використовується для управління потоком інформації в реальному часі і на рівні підприємства, що дозволяє своєчасно приймати рішення.

Обробка складних подій.

Обробка складних подій (Complex event processing (CEP)) дозволяє за шаблонами простих і звичайних подій проводити аналіз того, чи наступила складна подія. Обробка складних подій полягає в оцінюванні взаємного впливу

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		47

подій і в наступному виконанні дій. При цьому, типи подій (відомих або звичайних) можуть перетинатись, а події можуть виникати протягом тривалого періоду часу.

Кореляція подій може бути причинною, тимчасовою або просторовою. СЕР вимагає використання складних інтерпретаторів подій, визначення і підбору шаблонів подій, а також відповідних кореляційних методів. Обробка складних подій зазвичай використовується для виявлення і реагування на аномальну поведінку, загрози і можливості у бізнесі.

Jira – була використана комерційна система відслідковування помилок, призначена для організації взаємодії з користувачами, хоча в деяких випадках використовується і для управління проектами. Розроблено компанією Atlassian, є одним з двох її основних продуктів (поряд з вікі-системою Confluence). Має веб-інтерфейс.

Назва системи отримано шляхом усічення слова «Gojira» – Японського імені монстра Годзилла, що, в свою чергу, є відсиланням до назви конкуруючого продукту – Bugzilla; створювалася в якості заміни Bugzilla і багато в чому повторює її архітектуру. Система дозволяє працювати з декількома проектами. Для кожного з проектів створює і веде схеми безпеки і схеми оповіщення.

До версії 3.13.5 (включно) розрізнялися редакції Enterprise, Professional і Standard, після – Залишилася тільки редакція Enterprise (для великих організацій).

Система заснована на Java EE і працює на кількох популярних системах управління базами даних і операційних системах.

Основний елемент обліку в системі – завдання (ticket або issue). Завдання містить назву проекту, тему, тип, пріоритет, компоненти і зміст. Завдання може бути розширена додатковими полями (також і нові призначені для користувача поля можуть бути визначені), додатками (наприклад – Фотографіями, скріншотами) або коментарями. Завдання може редагуватися або просто змінювати статус, наприклад, з «відкритий» в «закритий».

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		48

Які переходи між станами можливі, визначається через настраюється потік операцій. Будь-які зміни в задачі записуються в журнал.

Jira має велику кількість можливостей конфігурації: для кожної програми може бути визначений окремий тип завдання з власним workflow, набором статусів, одним або декількома видами уявлення (screens). Крім того, за допомогою так званих «схем» можна визначити для кожного індивідуального Jira-проекту власні права доступу, поведінку і видимість полів і багато іншого.

Завдяки універсальному підходу можна пристосувати Jira для багатьох непрофільних завдань, наприклад, керування вимогами, керування ризиками, аж до реалізації невеликої системи бронювання, автоматизації процесу рекрутингу.

Для інтеграції з зовнішніми системами підтримує інтерфейси SOAP, XML-RPC і REST. Поставляється із засобами інтеграції з такими системами управління версіями, як Subversion, CVS, Git, Clearcase, Team Foundation Server, Mercurial і Perforce.

Існують доповнення, що дозволяють вбудувати Jira в інтегровані середовища розробки, в тому числі Eclipse і IntelliJ IDEA. Перекладена багатьма мовами, включаючи українську, англійську, японську, німецьку, французьку, іспанську.

Для сторонніх розробників надаються кошти розробки розширень системи – плагінів. Розробники розширень можуть викладати плагіни для продажу на спеціальний розділ сайту Atlassian.

Є комерційним продуктом, який може бути ліцензований для роботи на локальному сервері або доступний в якості віддаленого додатки. Ціноутворення залежить від максимального числа користувачів, при цьому близько \$50 за користувача для локального і \$7 на місяць за користувача для віддаленого доступу є типовими цінами.

Для академічних і комерційних клієнтів доступний повний вихідний код під ліцензією розробника.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		49

Для проектів з відкритим вихідним кодом Atlassian надає спеціальну безкоштовну ліцензію при дотриманні наступних правил:

- проект використовує ліцензії, схвалені Open Source Initiative;
- Вихідний код проекту доступний для скачування;
- у проекту є публічно доступна веб-сайт;
- програмне забезпечення від Atlassian є на веб-сайті проекту.

4.2 Захист розробленого програмного забезпечення

Розроблене програмне забезпечення захистимо за допомогою алгоритму захисту інформації RSA. Спочатку необхідно обчислити пару ключів (секретний ключ і відкритий ключ). Для цього відправник електронних документів обчислює два більших простих числа P і Q , потім знаходить їхній добуток $N = P * Q$ і значення функції $\varphi(N) = (P-1)(Q-1)$. Далі відправник обчислює число E з умов $E < \varphi(N)$, НЗД($E, \varphi(N)$) = 1 і число D з умов $D < N$, $E * D \equiv 1 \pmod{\varphi(N)}$.

Пари чисел (E, N) є відкритим ключем. Цю пару чисел автор передає партнерам по переписці для перевірки його цифрових підписів. Число D зберігається автором як секретний ключ для підписування.

Допустимо, що відправник хоче підписати повідомлення M перед його відправленням. Спочатку повідомлення M (блок інформації, файл, таблиця) стискають за допомогою геш-функції $h(-)$ у ціле число m : $m = h(M)$.

Потім обчислюють цифровий підпис S під електронним документом M , використовуючи геш-значення m і секретний ключ D : $S = m \pmod{N}$.

Пари (M, S) передається партнерові-одержувачеві як електронний документ M , підписаний цифровим підписом S , причому підпис S сформований власником секретного ключа D .

Після прийому пари (M, S) одержувач обчислює геш-значення повідомлення M двома різними способами. Насамперед, він відновлює геш-значення m' , застосовуючи криптографічне перетворення підпису S з

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		50

використанням відкритого ключа E : $m' = S^E \pmod{N}$.

Крім того, він знаходить результат гешування прийнятого повідомлення M з допомогою такої ж геш-функції $h(-)$: $m = h(M)$.

Якщо дотримується рівність обчислених значень, тобто $S^E \pmod{N} = h(M)$, то одержувач визнає пару (M, S) справжньою. Доведено, що тільки власник секретного ключа D може сформувати цифровий підпис S по документі M , а визначити секретне число D по відкритому числу E не легше, ніж розкласти модуль N на множники. Крім того, можна строго математично довести, що результат перевірки цифрового підпису S буде позитивним тільки в тому випадку, якщо при обчисленні S був використаний секретний ключ D , що відповідає відкритому ключу E . Тому відкритий ключ E іноді називають "ідентифікатором" того, хто підписав.

К6ПЗ-2026

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		51

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

Програма має простий та інтуїтивно зрозумілий інтерфейс, який зображений на рисунку 5.1. З рисунку головного вікна можна побачити що інтерфейс головного вікна розподілено на наступні функціональні розділи:

- Функціональних кнопок ПЗ: Формування даних сигналу керування; Статистичне навантаження; Аналіз даних; Налаштування; Звіт.
- Навігаційного меню яке викликається натисканням правої клавіші маніпулятора миші: Налаштування; Довідка.
- Верхнього меню: Мережа; Налаштування; Довідка.
- Розділу обрання групи перегляду.
- Розділу виведення результату роботи системи.

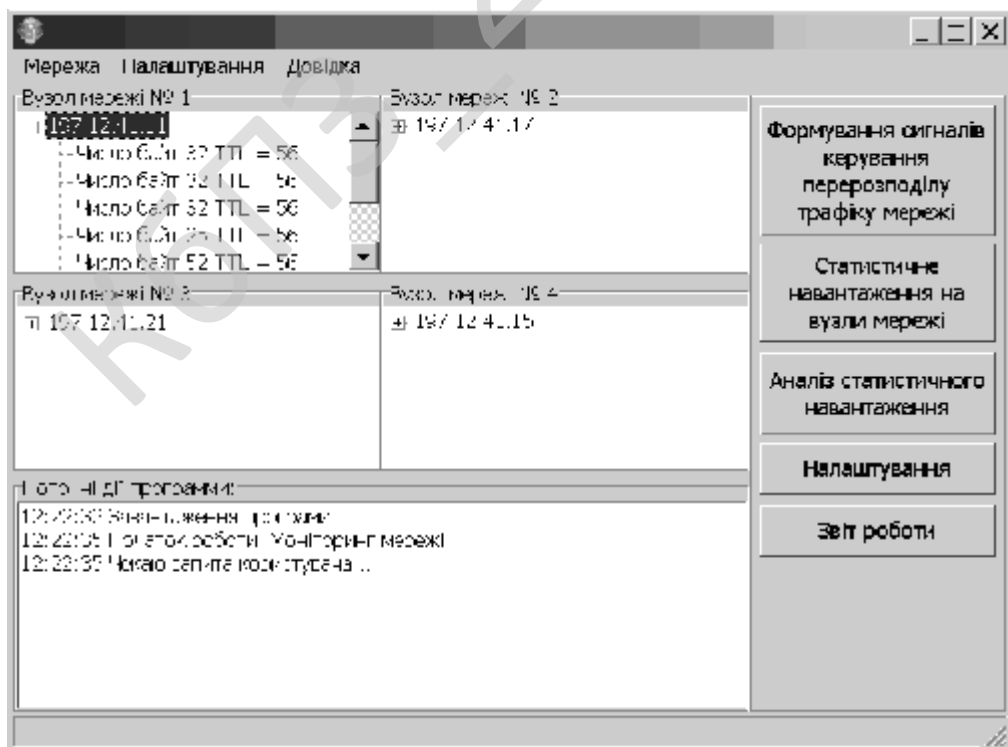


Рисунок 5.1 – Головне вікно ПЗ

На рисунку 5.2 зображено авторські дані розробленого програмного забезпечення. Розроблена програма має дуже простий і зрозумілий інтерфейс з користувачем. Кожен, хто в достатньому обсязі володіє операційним середовищем Windows без особливих складностей освоїть і цю програму, оскільки її інтерфейс інтуїтивно зрозумілий. Якщо програма не видала ніяких помилок, і працює, то можна використовувати, інакше слід слідувати інструкціям, які пропонує програма.

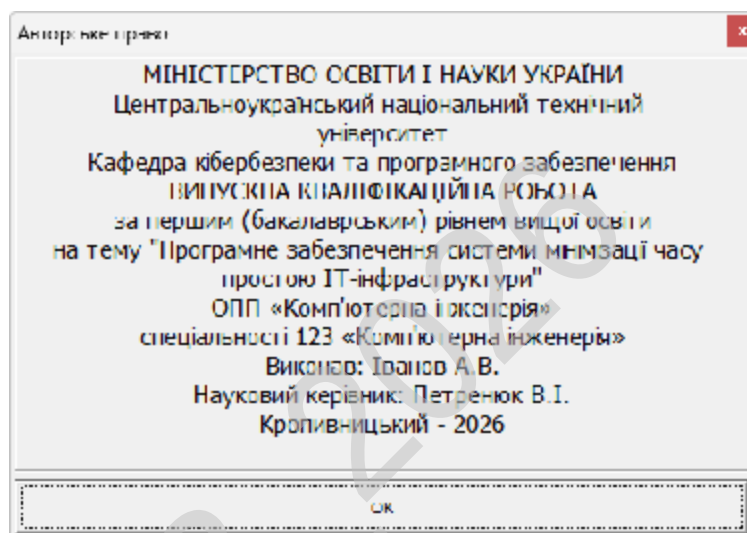


Рисунок 5.2 – Авторське право

Процес впровадження програмного забезпечення, це процес налаштування програмного забезпечення під певні умови використання, а також навчання користувачів роботі з програмним продуктом. Впровадження програмного забезпечення це усі дії, що роблять розроблену програмну систему готовою до використання. Даний процес є частинною життєвого циклу програмного забезпечення. Таким чином у результаті вищерозглянутого можна стверджувати що розроблено інтерфейс системи у відповідності з вибраною метою роботи. Система містить максимальний необхідний набір функцій придатних для виконання будь-яких дій для забезпечення повноцінної роботи програми. Далі розглянемо висновки та використані літературні джерела.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		53

6 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти, призначено для системи мінімізації часу простою ІТ-інфраструктури.

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

Рішення завдання полягало у вирішенні наступних задач:

- Був проведений огляд існуючих систем мінімізації часу простою ІТ-інфраструктури.
- Досліджена система мінімізації часу простою ІТ-інфраструктури.
- На основі отриманих результатів досліджень створена програмна реалізація системи мінімізації часу простою ІТ-інфраструктури.

Розроблені під час виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання мінімізації часу простою ІТ-інфраструктури.

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 мінімізації часу простою ІТ-інфраструктури. Це дозволило мінімізувати строк розробки програмного забезпечення, і, як слід, 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 RSA.

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.

К6ПЗ-2026

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		55

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Peter Hoddie, Lizzie Prader «IoT Development for ESP32 and ESP8266 with JavaScript: A Practical Guide to XS and the Moddable SDK» ISBN-13 (pbk): 978-1-4842-5069-3 ISBN-13 (electronic): 978-1-4842-5070-9

2. STM32CubeMX for STM32 configuration and initialization C code generation. User manual. June 2022. 397 p.

3. І.В.Чихіра, А.Г. Микитишин Конспект лекцій з дисципліни «Програмування систем реального часу» / Укладачі : Чихіра І.В., Микитишин А.Г., – Тернопіль : Тернопільський національний технічний університет імені Івана Пулюя , 2016. – 76 с.

4. Jack Ganssle and Michael Barr. 2003. Embedded Systems Dictionary. CMP Books.

5. Технології інтернету речей. Навчальний посібник [Електронний ресурс]: / Б. Ю. Жураковський, І.О. Зенів; КПП ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 12,5 Мбайт). – Київ: КПП ім. Ігоря Сікорського, 2021. – 271 с.

6. Greg Dunko, Joydeep Misra, Josh Robertson, Tom Snyder “A reference guide to the Internet of Things” / 2017 Bridgera LLC, RIoT.

7. Donald Norris “Programming with STM32. Getting started with the Nucleo Board and C/C++” 416 p. 2018.

8. Neil Kolban “Kolban’s book on ESP32”. Texas, USA. 951 p.

9. Вінтенко Б.Ю., Миронець І.В., Смірнов О.А. «Модель комп’ютерно-орієнтованої процедури системи підтримки оперативного персоналу АЕС». *IV Міжнародна науково-практична Інтернет-конференція «Інновації та перспективні шляхи розвитку інформаційних технологій (ІПШРІТ-2025)»* м.Черкаси 25 листопада 2025 року – Черкаси: ЧДТУ.– 2025. – С.101-103.

					ВКРБ-123.26.0039.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		56

10. Kuznetsov O., Frontoni E., Kuznetsova Y., Chevardin V., Smirnov O. «Architectural foundations for adaptive security in edge computing systems». *Cybersecurity Defensive Walls in Edge Computing*, 2025. pp. 21-61.

11. Вінтенко, Б.Ю., Миронець, І.В., Смірнов, О.А., Коваленко, О.В., Усік, П.С., Буравченко, К.О., Лисенко, І.А. «Логіко-структурна модель комп'ютерно-орієнтованої процедури системи підтримки оперативного персоналу АЕС». *Кібербезпека: освіта, наука, техніка*. 2025. Том 2 № 30. С. 413-427, 2025.

12. Смірнова, Т.В. «Дослідження методів, моделей та сучасних ІТ-рішень для підтримки технологічних процесів у критичній інфраструктурі держави». *Кібербезпека: освіта, наука, техніка*. 2025. Том 2 № 30. С.195-208, 2025.

13. Kuznetsov, O., Frontoni, E., Kuznetsova, K., Arnesano, M., Smirnov, O. «A secure biometric authentication architecture for blockchain-driven cyber-physical systems». *Security and Privacy of Cyber Physical Systems Emerging Trends Technologies and Applications*, 2025, pp. 193–224.

14. Kuznetsov, O., Atzeni, G., Arnesano, M., Randieri, C., Smirnov, O. «Secure IoT-based smart wheelchair system: From implementation to security enhancement strategy». *Security and Privacy of Cyber Physical Systems Emerging Trends Technologies and Applications*, 2025, pp. 225–257.

15. Kuznetsov, O., Smirnov, O., Kuznetsova, T., Shaikhanova, A., Svatowsky, I. «Privacy-utility trade-offs in IoT networks: A comparative analysis of differential privacy mechanisms for sensor data aggregation». *Security and Privacy of Cyber Physical Systems Emerging Trends Technologies and Applications*, 2025, pp. 589–622.

16. 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.

17. 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.

18. Вінтенко Б.Ю., Смірнов О.А., Миронець І.В., Смірнова Т.В., Коваленко О.В., Мацуй А.М. «Модель шляхів отримання вхідних даних комп'ютерної інтелектуальної системи підтримки оперативного персоналу АЕС». *Центральноукраїнський науковий вісник. Технічні науки*. 2025. Вип. 11(42), ч. II. С.52-62.

19. Вінтенко Б.Ю., Смірнов О.А., Миронець І.В., Смірнова Т.В. «Методи забезпечення відмовостійкості інтелектуальних систем підтримки оператора». *VIII міжнародна науково-практична конференція “Інформаційна безпека та комп'ютерні технології”*, м. Кропивницький. 24-25 квітня 2025 р. – Кропивницький: ЦНТУ. – 2025. – С. 44-46.

20. Вінтенко, Б., Миронець, І., Смірнов, О., Коваленко, А., Коноплицька-Слободенюк, О., Смірнова, Т., Константинова, Л. «Дослідження застосування систем підтримки оперативного персоналу об'єкту критичної інфраструктури при керуванні енергоблоком АЕС з реактором типу ВВЕР-1000». *Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка»*, 2024. № 2(26), С. 6-26.

21. 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.

22. Вінтенко, Б., Миронець, І., Смірнов, О., Кравчук, О., Козірова, Н., Савеленко, Г., Коваленко, А. «Дослідження вимог та аналіз кібербезпеки

програмного забезпечення інформаційно-керуючих систем АЕС, важливих для безпеки». *Кібербезпека: освіта, наука, техніка*. 2024. №3(23), С. 111-131.

23. 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.

24. Kuznetsov, O., Kuznetsova, Y., Smirnov, O., Kostenko, O., Zvieriev, V. «Evaluating Hashing Algorithms in the Age of ASIC Resistance». *CEUR Workshop Proceedings*, 2023, 3628, pp. 93-105.

25. 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.

26. 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.

27. Вінтенко Б.Ю., Смірнов О.А., Коваленко А.С., Смірнов С.А., Буравченко К.О. «Дослідження вимог міжнародних стандартів IEC60880 та IEC62138 з розробки програмного забезпечення інформаційно-керуючих систем АЕС, важливих для безпеки». *Системи управління, навігації та зв'язку*, 2023, вип. 3(73), С. 155-166.

28. 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.

29. 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.

30. 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.

31. 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.

32. 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.

33. 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.

34. 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.

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

36. 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.

37. 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.

38. 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.

39. 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.

40. 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.

41. 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.

42. 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.

43. Smirnov, O., Kuznetsov, A., Kolovanova, I., Kuznetsova, T., «Noise immunity of the algebraic geometric codes». *International Journal of Computing*; 2019,

44. 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.

45. 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.

46. 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.

47. 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.

48. 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.

49. 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.

50. 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.

51. 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.

52. 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

53. Смірнов О.А., Усік П.С., Миронець І.В., Буравченко К.О., Якименко Н.М. «Метод підвищення ефективності розподіленої обробки даних у комп'ютерних системах операторів стільникового зв'язку» *Вісник Черкаського державного технологічного університету. Технічні науки*. №4. С. 103-110. 2020.

54. 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.

55. 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.