**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**
**ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»**

*Спеціальність: 123 «Комп'ютерна інженерія»*

*Освітньо-професійна програма: «Обслуговування комп'ютерних систем і мереж»*

*Група: 4КС-57*

# Дипломний проект

**здобувача освіти денної форми навчання**
**КС.57.02.000.ДП**

***ВАСИЛЬЄВА***
***ВАДИМА СЕРГІЙОВИЧА***

**м. Одеса**
**2024 р.**

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: **123 «Комп'ютерна інженерія»**

Освітньо-професійна програма: «**Обслуговування комп'ютерних систем і мереж»**

Група: **4КС-57**

# ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту на тему:

## Розробка програмних засобів діагностики вузлів комп'ютерної мережі за протоколом ICMP

Проектний матеріал складається з пояснювальної записки на 84 сторінках та графічного (презентаційного) матеріалу на 15 аркушах (слайдах)

Дипломник _____ ( Васильєв В.С.)
Керівник _____ ( Кривченко А.А.)

**Консультанти:**

з економічного розділу _____ ( Іванченков В.С.)
з розділу охорони праці та техніки безпеки _____ ( Чорновол Н.І.)
з нормоконтролю _____ ( Петрашова В.І.)
старший консультант _____ ( Кривченко Ю.В.)

**До захисту допущений**

Голова циклової комісії _____ ( Кривченко Ю.В.)
Завідувач відділення _____ ( Скорнякова О.В.)

Захист «17» 06 2024 р. Протокол ЕК № 1
Оцінка ЕК 4 (добре) 76 з.

Секретар ЕК _____

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»**

Відділення *комп'ютерних систем* Комісія *КТ та ПІ*

Спеціальність *123 «Комп'ютерна інженерія»*

Освітньо-професійна програма *«Обслуговування комп'ютерних систем і мереж»*

ЗАТВЕРДЖУЮ:

Заст. дир. з НВР *Беркань І.В.*

"15" 01 2024 р.

# ЗАВДАННЯ

**на дипломний проект**

Здобувачеві освіти *Васильєву Вадиму Сергійовичу*

(прізвище, ім'я, по батькові)

1. Тема проекту *Розробка програмних засобів діагностики вузлів комп'ютерної мережі за протоколом ICMP*

затверджена наказом по коледжу від "02" 11 2023 р. № 244-А2-ОД

2. Термін здачі закінченого проекту 10.06.2024

3. Вихідні данні до проекту *Реалізувати пошук нестабільно працюючих станцій локальної обчислювальної мережі; У складі тестової мережі виконати діагностику 21 робочої станції (20 PC + 1 сервер); Специфікації протоколу ICMP; 4. Специфікації процедури ping; Застосунок повинен мати візуальний інтерфейс під ОС Windows з можливістю виконання необхідних налаштувань за допомогою списків вибору та перемикачів*

4. Зміст розрахунково-пояснювальної записки (перелік питань, які необхідно розробити)

*Огляд систем мережевого моніторингу*

*Аналіз вимог до системи мережевого моніторингу*

*Вибір засобів та інструментів реалізації програмного забезпечення*

*Розробка ПЗ діагностики вузлів комп'ютерної мережі за протоколом ICMP*

*Тестування розробленого застосунку для визначеної конфігурації комп'ютерної мережі*

5. Перелік графічного (презентаційного) матеріалу (з точним зазначенням обов'язкових креслень, кількості слайдів)

*Комутована мережа з аналізатором протоколів; Аналізатор протоколів і моніторинг мережевого трафіку; Узагальнена схема структури мережі; Схема стенду для тестування ПЗ; Узагальнений алгоритм пошуку несправностей у мережі; Розширений алгоритм діагностики мережі; Алгоритм завдання початкових параметрів; Алгоритм роботи процедури Ping; Головне вікно застосунку діагностики вузлів комп'ютерної мережі; Побудова мережі для випробування розробленого застосунку; Графіки стану абонентів у мережі*

6. Консультанти по проекту, із зазначенням розділів проекту, що їх стосується

| Розділ | Консультант | Підпис, дата | |
|---|---|---|---|
| | | Завдання видав | Завдання прийняв |
| Основний розділ | *Кривченко А.А.* | | |
| Економічний розділ | *Іванченков В.С.* | | |
| Розділ охорони праці | *Чорновол Н.І.* | | |
| Нормоконтроль | *Петрашова В.І.* | | |
| Старший консультант | *Кривченко Ю.В.* | | |

7. Дата видачі завдання 15.01.2024

Керівник *Кривченко А.А.* _____
(підпис)

Завдання прийняв до виконання *Васильєв В.С.* _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

| № з/р | Назва етапів дипломного проекту | Термін виконання етапів дипломного проекту (роботи) | Відмітка про виконання |
|---|---|---|---|
| 1 | *Вступ. Постановка мети та задач проектування* | 29.04.2024 | виконано |
| 2 | *Обґрунтування доцільності розробки застосунку* | 19.05.2024 | виконано |
| 3 | *Аналітичний огляд систем мережевого моніторингу* | 20.05.2024 | виконано |
| 4 | *Аналіз вимог до системи мережевого моніторингу* | 24.05.2024 | виконано |
| 5 | *Вибір технічних та програмних засобів розробки* | 25.05.2024 | виконано |
| 6 | *Опис ядра системи мережевого моніторингу* | 26.05.2024 | виконано |
| 7 | *Опис бібліотек для роботи з мережевими пристроями* | 28.05.2024 | виконано |
| 8 | *Розробка алгоритмів та структури додатку* | 30.05.2024 | виконано |
| 9 | *Розробка ПЗ діагностики вузлів комп'ютерної мережі* | 01.06.2024 | виконано |
| 10 | *Створення та налагодження стенду тестування ПЗ* | 02.06.2024 | виконано |
| 11 | *Випробування роботи ПЗ у мережі* | 03.06.2024 | виконано |
| 12 | *Аналіз результатів, підготовка слайдів презентації* | 05.06.2024 | виконано |
| 13 | *Економічні розрахунки та питання з охорони праці* | 06.06.2024 | виконано |
| 14 | *Підготовка графічної частини проекту* | 07.06.2024 | виконано |
| 15 | *Підготовка проекту до захисту та тестування ПЗ* | 10.06.2024 | виконано |

Дипломник _____
(підпис)

Керівник _____
(підпис)

| Формат | Зона | Поз. | Позначення | Назва | Кіл. | Примітка |
|---|---|---|---|---|---|---|
| | | | | Документація | | |
| | | | КС 57. 02 000. 00 ДП | Дипломний проект | | |
| А4 | | | КС 57. 02 000. 00 ДП ПЗ | Пояснювальна записка | 1 | |

| Зм. | Арк. | № докум. | Підпис | Дата | КС 57. 02 000. 00 ДП | | |
|---|---|---|---|---|---|---|---|
| Розробив | | Васильєв В.С. | | | Розробка програмних засобів діагностики вузлів комп'ютерної мережі за протоколом ICMP | Літ. | Аркуш 4 / Аркушів 84 |
| Перевірив | | Кривченко А.А. | | 10.06 | | Н Д П | |
| Н. контр. | | Петрашова В.І. | | 10.06 | | ВСП "ОТФК ОНТУ" гр.4КС-57 | |
| Затверд. | | Кривченко Ю.В. | | 10.06 | | | |

# ЗМІСТ

# ВСТУП

Інформаційна інфраструктура сучасного підприємства представляє собою складний конгломерат різномасштабних та різнорідних комутацій і систем, тім моніторинг мережі, облік потоку та розмежування доступа у глобальну мережу стає серйозним та складним завданням.

Існують спеціальні стандарти моделей керування мережами, що дозволяють полегшити керування великими різнорідними мережами. Задля керування мережами передачі інформації у правила IP існує протокол керування системою SNMP (Simple Network Manager Protocol). Таким чином, протокол SNMP передбачає керувати різними об'єктами мережі централізовано.

Нажаль, стандартні інструменти MS Windows не володіють необхідною функціональністю задля вирішення подібного роду задач. Тім, виходом із такої ситуації є розробка спеціалізованого програмованого середовища.

Дипломна робота присвячена діагностиці обчислювальної мережі, знаходженню її несправних чи недоступних сегментів чи сегментів. В даній роботі буде проведено аналіз обчислювальної мережі із точки зору появи у ній недоступних робочих станцій чи виходу тих із ладу. Разом з цим буде запропонований спосіб автоматичного пошуку збійних сегментів мережі за поміччю спеціально складеної додатку, котра значно полегшує роботу системним адміністраторам.

Розроблювані програмні інструменти діагностування сегментів обчислювальної мережі за протоколом I-C-M-P мають допомагати проводити моніторинг внутрішньої обчислювальної мережі на появу у ній помилок. Програма буде проводити дослідження доступа до усіх робочих станцій і серверів мережі і записувати дані цих результатів в спеціальний файл. Оцінюючи занесені дані програма містить проводити перевірку у доступа і дозволяти визначити дані щодо період функціювання, призупинення чи збій будь-якого елемента мережі. Є дозволить системному адміністратору не витрачати зайвого терміну на виявлення причини несправності мережі, але одразу приступити до розв'язання задачі.

# 1 ОСНОВНА ЧАСТИНА

## 1.1 Аналітичний огляд систем комп'ютерного відстежування

Розглянемо деякі актуальні задачі, що виникають в комп'ютерних адміністраторів підприємств.

В традиційній внутрішньої мережі із середовищем передачі, що розділяється, (рис.1.1.) визначник правил чи монітор спроможне досліджувати весь потік даного сегмента мережі. Хоча перевагу комутованої мережі у продуктивності іноді майже не помітно, поширення комутованих архітектур мало катастрофічні наслідки задля традиційних систем діагностування.

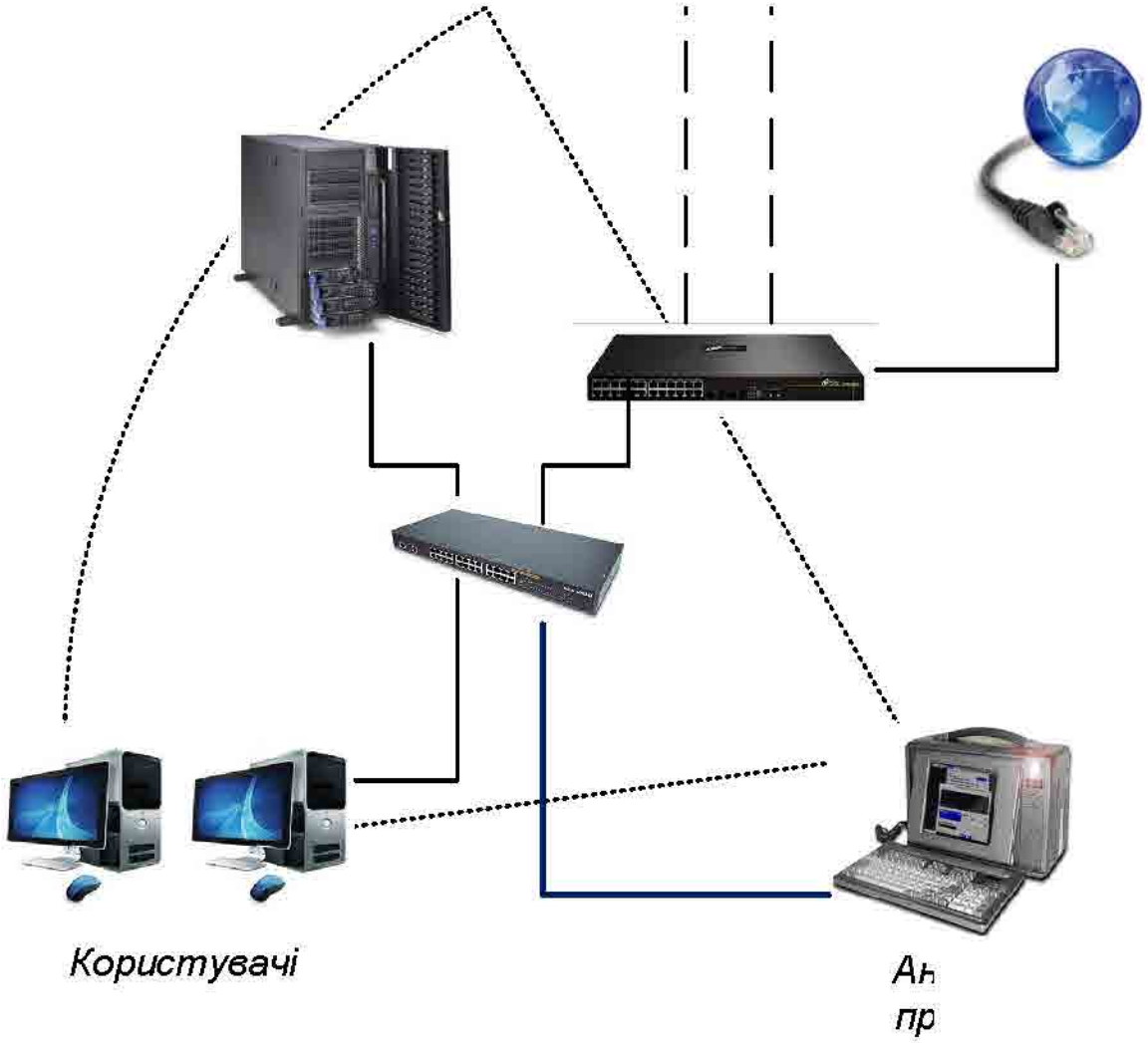



Рисунок 1.1. Традиційна локальна мережа із середовищем передачі, що розділяється, та аналізатором правил

В сильно сегментованій мережі аналізатори правил здатні бачити лише одноадресний потік на окремому порту комутатора, на відміну з мережі колишньої топології, де вони могли ретельно дослідити будь-котрий пакет у домені колізій. У таких умовах традиційні інструменти відстежування не спроможні зібрати статистику у усіх «діалогах», тім що кожна пара кінцевих точок, котрі «перемовляються», користується, у суті, своєю власною системою.

У комутованої мережі (рис.1.2) визначник правил у одній точці спроможне «бачити» лише єдиний сегмент, коли комутатор не здатний дзеркально відображати кілька портів одночасно.

Задля збереження контролю над сильно сегментованими мережами виробники комутаторів пропонують різноманітні інструменти задля відновлення повної «видимості» мережі, однак на цьому шляху залишається чимало труднощів. В комутаторах, що поставляються зараз, зазвичай підтримується «дзеркальне відображення» портів, коли потік одного із тих дублюється на раніше незадіяний порт, до якого підключається монітор чи визначник.

Однак «дзеркальне відображення» містить низку недоліків. У-перше, у кожен момент терміну видно лише один порт, тім виявити неполадки, що зачіпають відразу декілька портів, дуже непросто. У-друге, дзеркальне відображення спроможне привести до зниження продуктивності комутатора. У-третє, на дзеркальному порту зазвичай не відтворюються збої фізичного рівня, але іноді навіть губляться позначення уявних локальних комутацій. Нарешті, в багатьох випадках не спроможні у повній мірі дзеркально відображатися повнодуплексні канали Ethernet.

Таким чином, проблема діагностування комп'ютерних комутацій є актуальною та у кінцевому рахунку, діагностування несправностей є задачею керування. Задля більшості критично важливих корпоративних систем проведення тривалих відновлювальних робіт не допустиме, тім єдиним рішенням буде застосування резервних систем та процесів, здатних взяти на себе необхідні процедури негайно після виникнення збоїв.

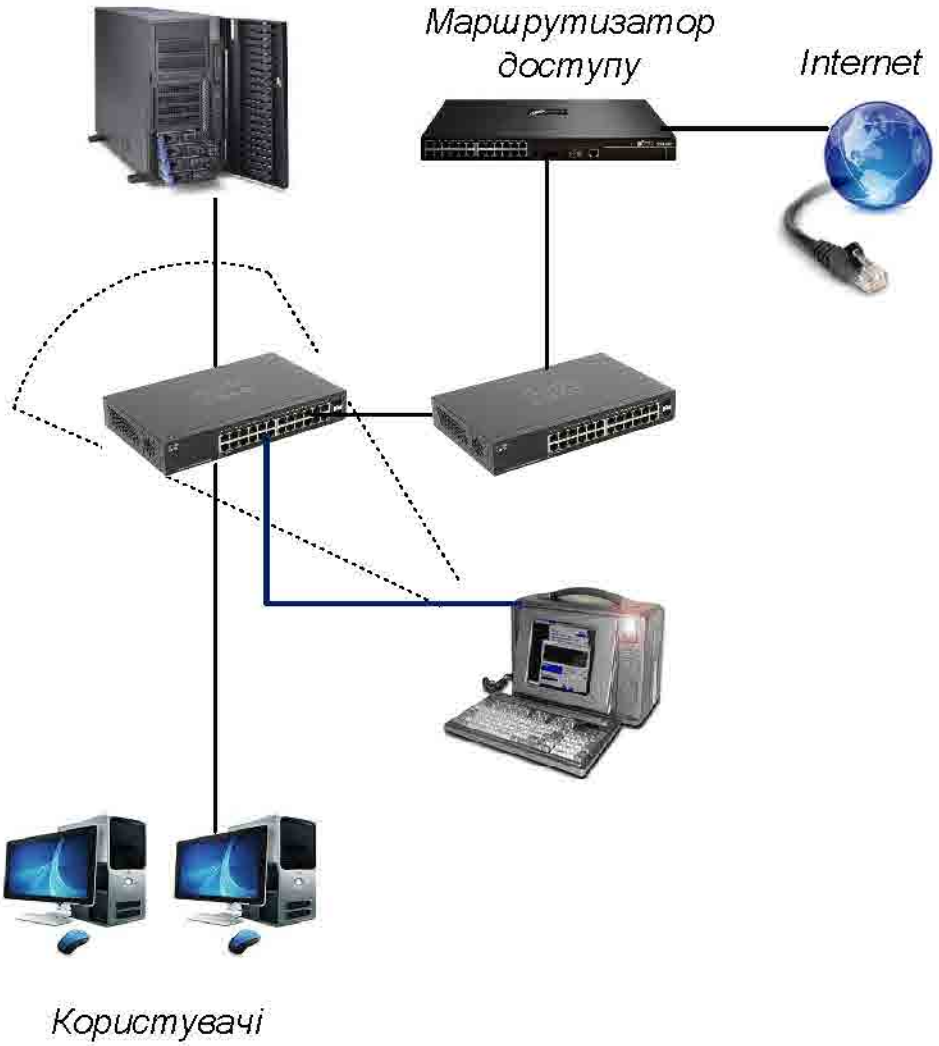



Рисунок 1.2. Комутована мережа із аналізатором правил

### 1.1.1 Існуючі програмні інструменти діагностування

Серед комп'ютерних систем діагностування комп'ютерних комутацій, можливо виділити спеціальні моделі керування системою (Network Management Systems) – централізовані програмні моделі, котрі збирають дані щодо стан сегментів та комунікаційних систем мережі, але разом з цим дані щодо потік, циркулюючий у мережі. Ці моделі не лише здійснюють моніторинг та аналіз мережі, але та виконують у автоматичному чи напівавтоматичному режимі дії із керування системою – включення та відключення портів систем, зміна параметрів мостів адресних таблиць мостів, комутаторів та маршрутизаторів та т.п. Прикладами систем керування спроможні служити популярні моделі HPOpenView, SunNetManager, IBMNetView.

Інструменти керування системою (System Management) виконують процедури, аналогічні функціям систем керування, але у відношенню до комунікаційного устаткування. Разом із тим, деякі процедури цих двох видів систем керування спроможні дублюватися, зокрема, інструменти керування системою спроможні виконувати найпростіший аналіз комп'ютерного потоку.

Експертні моделі акумулюють людські знання щодо виявлення причин аномальної функціювання комутацій та можливі способи приведення мережі у працездатний стан. Експертні моделі часто реалізуються в вигляді окремих підсистем різних систем відстежування і визначення комутацій: систем керування мережами, аналізаторів правил, комп'ютерних аналізаторів. Найпростішим варіантом експертної моделі є контекстно-залежна help-структура. Більше складні експертні моделі являють собою так звані бази знань, що володіють елементами штучного інтелекту. Прикладом такої моделі є експертна структура, вбудована у систему керування Spectrum компанії Cabletron.

### 1.1.2 Застосування аналізаторів правил

Визначник правил – життєво важлива частина набору інструментів комп'ютерного адміністратора. Щоб дізнатися, чому будь-котрий мережевий пристрій не працює саме так, але не інакше, слід використовувати визначник правил, щоб «вживу» відчути потік та виділити із нього дані та протоколи, що передаються із комп'ютерного кабелю. Визначник комп'ютерних правил спроможне використовуватися задля:

- локалізації складних проблем;
- виявлення і ідентифікації недозволеного ПЗ;
- отримання такої інформації, як базові моделі потоку (baseline traffic patterns) та метрики утилізації мережі;
- ідентифікації невикористовуваних правил задля видалення тих із мережі;
- генерації потоку задля випробування на вторгнення (penetration test) із

метою перевірки моделі захисту;

- функціювання із системами виявлення вторгнень Intrusion Detection System (IDS);
- прослуховування потоку, тобто локалізації недозволеного потоку із використанням Instant Messaging (IM) чи бездротових точок доступа Access Point – (AP);
- вивчення функціювання мережі.

Визначник правил є чи самостійним спеціалізованим пристроєм, чи персональним комп'ютером, зазвичай переносним, оснащеним спеціальною мережевою картою та відповідним програмним забезпеченням. Застосовувані мережева карта та програмне середовища повинні відповідати топології мережі (кільце, шина, зірка). Визначник підключається до мережі точно так само, як та звичайний вузол (рис.1.3). Відмінність полягає у тім, що визначник спроможне приймати всі пакети інформації, що передаються у мережі, у той період як звичайна станція – лише адресовані їй. Програмне середовища аналізатора складається із ядра, що підтримує роботу мережного адаптера та декодуючого одержувані дані, та додаткового програмованого коду, що залежить з типу топології досліджуваної мережі. До складу деяких аналізаторів спроможне входити разом з цим експертна структура, котра спроможне видавати користувачу рекомендації щодо те, котрі експерименти слід проводити у даній ситуації, що спроможні означати ті чи інші результати вимірювань, як усунути деякі види несправності мережі.

Методологія проведення визначення спроможне знаходитись представлена в вигляді наступних шести етапів:

- Захоплення інформації;
- Перегляд захоплених інформації;
- Пошук помилок (більшість аналізаторів полегшують цю роботу, визначаючи типи помилок та ідентифікуючи станцію, з якої прийшов пакет із помилкою);
- Дослідження продуктивності. Розраховується коефіцієнт застосування

пропускної здатності мережі чи середній період реакції на запит;

- Докладне дослідження окремих ділянок мережі. Зміст такого етапу конкретизується в міру того, як проводиться аналіз.

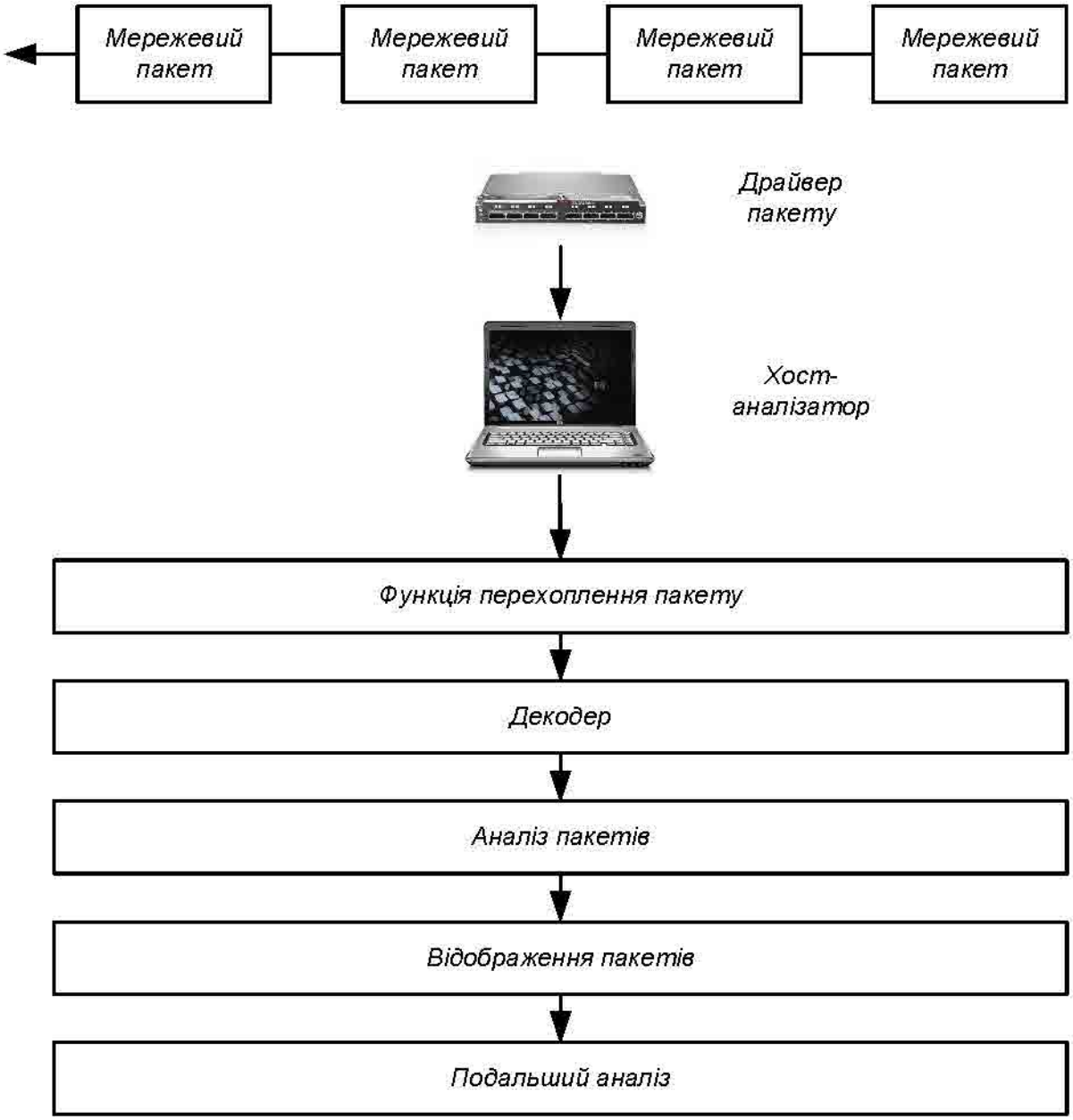



Рисунок 1.3. Визначник правил виконує моніторинг комп'ютерного потоку

Зазвичай хід визначення правил займає відносно небагато терміну – 1-2 робочих дня. Таким чином, сучасні аналізатори правил WAN/LAN/ATM дозволяють виявити помилки у конфігурації маршрутизаторів та мостів; встановити тип потоку, що пересилається із глобальної мережі; визначити використовуваний діапазон швидкостей, оптимізувати співвідношення поміж

пропускною спроможністю та кількістю каналів; здійснити повний моніторинг та декодування основних правил із будь-якого каналу; аналізувати статистику у реальному часі, включаючи аналіз потоку локальних комутацій через глобальні мережі.

### 1.1.3 Застосування правил відстежування

Протокол SNMP (англ. Simple Network Management Protocol – простий протокол керування системою) – є протокол керування мережами зв'язку на основі архітектури TCP/IP. Технологія покликана забезпечити керування та контроль за пристроями та додатками у мережі зв'язку шляхом обміну керуючою інформацією поміж агентами, що розташовуються на комп'ютерних пристроях, та менеджерами, розташованими на станціях керування. SNMP визначає мережу як сукупність комп'ютерних керуючих станцій та елементів мережі (головні машини, шлюзи та маршрутизатори, термінальні сервери), котрі спільно забезпечують адміністративні зв'язки поміж мережевими керуючими станціями та мережевими агентами.

При використанні SNMP присутні керовані і керуючі моделі. До складу керованої моделі входить компонент, званий агентом, котрий відправляє звіти керуючій системі. У суті SNMP-агенти передають керуючу дані на керуючі моделі як змінні (такі, як «вільна пам'ять», «ім'я моделі», «кількість працюючих процесів»).

До появи RMON протокол SNMP не міг використовуватися віддаленим чином, він допускав лише локальне керування пристроями. База RMON MIB володіє поліпшеним набором властивостей задля віддаленого керування, адже містить агреговану дані щодо пристрій, що не вимагає передачі у мережі великих обсягів інформації. Об'єкти RMON MIB включають додаткові лічильники помилок у пакетах, гнучкіші інструменти визначення графічних трендів та статистики, більше потужні інструменти фільтрації задля захоплення та визначення окремих наборів, але разом з цим більше складні умови встановлення сигналів попередження. Агенти RMON MIB більше інтелектуальні

порівняно із агентами MIB-I чи MIB-II та виконують значну частину функціювання у обробці інформації щодо пристрій, яку раніше виконували менеджери. Ці агенти спроможні розташовуватися всередині різних комунікаційних систем, але разом з цим знаходитись виконані в вигляді окремих комп'ютерних модулів, що працюють на універсальних PC та ноутбуках (прикладом спроможне служити LANalyzerHvell).

Інтелект агентів RMON передбачає їм виконувати прості дії у діагностиці несправностей та попередження щодо можливі відмови – зокрема, в рамках технології RMON можливо зібрати дані щодо нормальне функціювання мережі (тобто виконати так званий baselining), але потім виставляти попереджувальні сигнали, коли режим функціювання мережі відхилиться з baseline – є спроможне свідчити, зокрема, щодо неповну справність устаткування. Зібравши воєдино дані, що отримується з агентів RMON, додаток керування спроможне допомогти адміністратору мережі (що знаходиться, зокрема, за тисячі кілометрів з аналізованого сегмента мережі) локалізувати несправність та виробити оптимальний план дій задля її усунення.

Збір інформації RMON здійснюється апаратно-програмними зондами, що підключаються безпосередньо до мережі. Щоб виконати задачу збору та первинного визначення інформації, зонд повинен мати достатні обчислювальні ресурси та об'ємом оперативної пам'яті. У даний період на ринку є зонди трьох типів: вбудовані, зонди на базі комп'ютера та автономні. Продукт вважається підтримуючим RMON, коли у ньому реалізована хоча б одна група RMON. Зрозуміло, чим більше груп інформації RMON реалізовано у даному продукті, тим він, із одного боку, дорожче, але із іншого – тим більше повну дані щодо роботу мережі він надає.

Відмінною рисою стандарту RMON MIB є того незалежність з правила комп'ютерного рівня (на відміну з стандартів MIB-I та MIB-II, орієнтованих на протоколи TCP/IP). Тім, того зручно використовувати у гетерогенних середовищах, що використовують різні протоколи комп'ютерного рівня.

### 1.1.4 Існуючі апаратно-програмні моделі керування мережами

Структура керування системою (Network management system) – апаратні та/чи програмні інструменти задля відстежування і керування вузлами мережі. Програмне середовища моделі керування системою складається із агентів, що локалізуються на комп'ютерних пристроях та передають дані мережевій керуючій платформі. Метод інформаційного обміну поміж керуючими додатками та агентами на пристроях визначається протоколами.

Моделі керування мережами повинні мати цілу низку якостей:

- істинну розподіленість відповідно до концепції клієнт/сервер,
- масштабованість,
- відкритість, що передбачає впоратися із різнорідним (з настільних комп'ютерів до мейнфреймів) обладнанням.

Перші дві властивості тісно пов'язані. Добра масштабованість досягається за рахунок розподіленості моделі керування. Розподіленість означає, що структура спроможне включати кілька серверів та клієнтів. Сервери (менеджери) збирають дані щодо поточний стан мережі з агентів (SNMP, CMIP чи RMON), вбудованих у устаткування мережі, та накопичують тих в своїй базі інформації. Клієнти представляють собою графічні консолі, за якими працюють адміністратори мережі. Програмне середовища клієнта моделі керування приймає запити на виконання будь-яких дій з адміністратора (зокрема, побудова докладної карти частини мережі) та звертається за необхідною інформацією до сервера. Коли сервер містить потрібну дані, то він відразу ж передає її клієнту, коли немає – то намагається зібрати її з агентів.

При побудові карти мережі, котра складається із устаткування інших виробників, ці моделі починають помилятися та приймати одні пристрої за інші, але при керуванні цими пристроями підтримують лише тих основні процедури, але багато корисних додаткових функцій, котрі відрізняють даний пристрій з інших, структура керування просто не розуміє та, тім, не спроможне ними скористатися.

Задля виправлення такого недоліку розробники систем керування

включають підтримку не лише стандартних баз MIB I, MIB II та RMON MIB, але й численних приватних MIB фірм-виробників. Лідер у цій області – структура Spectrum, що підтримує близько 1000 баз MIB різних виробників.

Іншим способом більше якісної підтримки конкретної апаратури є застосування на основі будь-якої платформи керування додатку тієї фірми, котра випускає є устаткування. Провідні компанії – виробники комунікаційного устаткування – розробили та поставляють вельми складні та багатофункціональні моделі керування задля свого устаткування. Структура Optivity, зокрема, передбачає робити моніторинг та керувати мережами, що складаються із маршрутизаторів, комутаторів та концентраторів компанії BayNetwork, повністю використовуючи всі тих можливості та властивості. Устаткування інших виробників підтримується на рівні базових функцій керування. Структура Optivity працює на платформах OpenView компанії Hewlett-Packard та SunNetManager (попередник Solstice) компанії SunSoft. Однак, робота на основі будь-якої платформи керування із декількома системами, такими як Optivity, занадто складна та вимагає, щоб комп'ютери, на яких все є буде працювати, мали дуже потужні процесори та великі обсяги оперативної пам'яті [9].

Проте, коли у мережі переважає устаткування з будь-якого одного виробника, то наявність додатків керування такого виробника задля будь-якої популярної платформи керування передбачає адміністраторам мережі успішно вирішувати багато задач, тім розробники платформ керування поставляють разом із ними інструментальні інструменти, що спрощують розробку додатків, але наявність таких додатків та тих кількість вважаються дуже важливим фактором при виборі платформи керування.

## 1.2 Аналіз вимог до моделі діагностування сегментів обчислювальної мережі

Робота присвячена розробці комп'ютерних систем діагностування сегментів обчислювальної мережі за протоколом I-C-M-P але разом з цим дослідженню процесів, що протікають в локальній комп'ютерній мережі під дією

характерних несправностей, таких як, відсутність зв'язку із робочою станцією чи із сервером. Областю застосування розроблювання є сучасні обчислювальні моделі, що володіють високою продуктивністю, розширеним діапазоном подання чисел та достовірністю функціювання і об'єднані локальною обчислювальною системою, і потребують достовірності та точності передачі інформації.

Під період виконання розроблювання треба вивчити вплив несправностей в локальній обчислювальній мережі типу «відсутність доступа до абонента» і автоматизувати хід пошуку усіх доступних споживачів в мережі. Аналізується ймовірність появи помилки, що викликана характерною несправністю. Вивчається статистика появи помилок, викликаних характерними несправностями локальних обчислювальних мережах, що передбачає оцінювати достовірність обчислюваних результатів.

Розроблювана структура діагностування сегментів обчислювальної мережі містить допомогти визначати робочі і неробочі вузли внутрішньої обчислювальної мережі за поміччю системних команд (у принципу процедури пінг). Передбачаються режими задавання типів класів локальних обчислювальних комутацій, проміжних IP-адрес споживачів і серверів внутрішньої обчислювальної мережі, що потребують діагностування на отримання доступа до тих. Треба забезпечити спроможність виявлення недоступних споживачів локальних обчислювальних комутацій на протязі заданого проміжку терміну, що могло б призвести до фатальних несправностей в роботі великих обчислювальних комплексів. Задля такого треба, щоб розроблювана структура діагностування сегментів обчислювальної мережі проводила перевірка усіх чи заданих споживачів обчислювальної мережі на предмет тих доступа і функціювання задля заданих інтервалів терміну.

Вихідні дані та одержувані результати моделювання в розроблюваній системі діагностування сегментів обчислювальної мережі мають представлятися із використанням інформативної індикації. Зручність в використанні додатку містить ґрунтуватися на простоті та варіантності задавання режимів

функціювання, вихідних інформації, але разом з цим одержанні результатів моделювання в наочному вигляді.

Функціональність розроблюваної моделі діагностування сегментів обчислювальної мережі містить забезпечувати:

- задавання будь-яких, в тім числі довільних складових значень, їхній повний перебір та задавання послідовностей довільних значень;
- спроможність змінювати параметри і властивості заданої внутрішньої обчислювальної мережі;
- оцінку помилок та статистики їхнього виникнення.

## 1.3 Задачі комп'ютерного відстежування і методи тих вирішення

### 1.3.1 Розподіл прав доступа до внутрішніх та зовнішніх ресурсів та сервісів мережі

Внутрішня безпека мережі забезпечується використанням контролера домену. В кожного співробітника є свій обліковий запис задля входу у систему та свій пароль, котрий треба міняти раз у кілька місяців. В усіх співробітників чітко розмежовані права, зокрема, співробітники технічної підтримки не зможуть отримати доступ до баз інформації бухгалтера, відповідно бухгалтер, не зможе скористатися інформацією, із якою працює служба технічної підтримки.

Разом з цим стоїть задавання розмежування доступа до глобальної мережі користувачів корпоративної мережі, що можливо зробити на шлюзі – є сприятливо позначиться на роботі співробітників та убезпечить локальну мережу з шкідливого контенту та шкідливих додатків.

Разом з цим, на розсуд керівництва компанії, можливо обмежити доступ до деяких комп'ютерних сервісів, котрі спроможні використовуватися поза робочих цілей. Однак, задля більше гнучкої та швидкої зміни налаштувань обмеження без застосування ресурсів апаратних маршрутизаторів та буде реалізовано програмне середовища відстежування мережі і обліку потоку.

Задля того, щоб однозначно визначати конкретного споживача мережі,

буде здійснюватися перевірка так званих MAC-адреса – апаратна адреса комп'ютерного інтерфейса (комп'ютерної карти). Коли IP-адресу користувач спроможне змінити вручну, то MAC-адреса – є унікальний ідентифікатор конкретного комп'ютерного вузла, отже, та споживача мережі. Не пускаючи пакети із певною MAC-адресою далі в мережу, можливо обмежити доступ конкретного споживача у Internet чи віддаленому мережевому ресурсі. Тут першорядною проблемою є отримання вхідного потоку задля подальшого визначення. Вирішити її можливо «вставши» на мережевий вигляд та прослуховуючи того (прослуховуючи весь потік, котрий на нього надходить). Існує певна множина комп'ютерних бібліотек, котрі дають спроможність зчитувати мережевий потік. Вибір цих бібліотек буде розглянуто у наступному розділі. Другорядною задачею є розбір вхідних наборів задля вилучення із тих IP- та MAC-адрес, розміру наборів, але разом з цим іншої інформації. Вибір систем задля вирішення цієї задачі буде описано в наступному розділі.

### 1.3.2 Моніторинг активних систем

Моніторинг активного устаткування – одна із основних задач комп'ютерного монітора. У реалізованій програмі треба буде отримати дані щодо активні мережеві інтерфейси, котрі встановлені на локальному PC, адже саме із тих буде проводитися захоплення та аналіз наборів, але разом з цим тих подальша відправка.

Складання переліку активного мережного устаткування можливо отримати, аналізуючи MAC-адреси наборів, що надходять. Задля такого знадобляться спеціалізовані інструменти задля можливості прослуховування комп'ютерних інтерфейсів, але разом з цим задля «парсинга» (обробки) наборів, що надходять.

### 1.3.3 Відправлення отриманих наборів

Пакети, котрі надходять на приймаючий вигляд (будемо того називати клієнтським інтерфейсом) після тих визначення, підлягають подальшій відправці через другий вигляд (будемо того називати зовнішнім). Дану задачу можливо

вирішити написанням так званого програмованого маршрутизатора, але із деякими обмеженими можливостями.

Маршрутизатор («router») – спеціалізований мережевий комп'ютер, котрий містить два чи більше комп'ютерних інтерфейсів та пересилає пакети інформації поміж різними сегментами мережі. Задля прийняття рішень щодо пересилання наборів використовується інформація щодо топологію мережі та певні правила, задані адміністратором. Тобто є мережевий пристрій, котрий вміє передавати дані поміж різними мережами, зокрема системою інтернет-провайдера та корпоративною локальною системою.

Маршрутизатори працюють на більше високому «мережевому» (третьому) рівні комп'ютерної моделі OSI, ніж комутатор (чи мережевий міст) та концентратор (хаб), котрі працюють відповідно на другому та першому рівнях моделі OSI.

Існує ряд технологій у різних мовах програмування, на яких можливо реалізувати подібного роду роутер, але конкретна реалізація буде залежати з обраних систем програмування.

### 1.3.4 Зберігання інформації щодо мережеві пристрої

Задля функціювання програмованого середовища та виконання ним належних функцій, треба зберігати дані щодо мережеві пристрої. Зокрема, при закритті додатку, треба зберігати список дозволених MAC-адрес, прикріплених до тих IP-адрес, але разом з цим тих обсяг потоку. Варіантів задля вирішення цієї задачі – два. Перший – є застосування повноцінного серверу БД, зокрема, Microsoft SQL Server. Задля такого на локальній машині, де буде функціонувати ПЗ, що розробляється, треба встановити та налаштувати сервер БД, створити БД, таблиці та реалізувати логіку функціювання із цією БД. Другий варіант – застосування спеціальних призначених задля споживача файлів із певною структурою, котрі розташовуються у тій же директорії, що та виконавчий модуль додатку. Простота такого варіанту у тім, що задля функціювання із ними немає потреби у установці сервера БД. Робота із такими файлами здійснюється із

додатку безпосередньо.

В зв'язку із тим, що треба буде зберігати лише кілька властивостей комп'ютерних систем, потреба в використанні будь-якого сервера БД відпадає, але при розширенні функцій ПЗ у майбутньому застосування сервера спроможне знадобитися. Перейти з файлової структури зберігання інформації до повноцінної БД не складе труднощів.

## 1.4 Формування вимог до розроблюваних комп'ютерних систем

Вимоги до програмованого середовища – сукупність тверджень щодо атрибутів, властивостей чи якостей програмної моделі, котра підлягає реалізації.

Перелік загальних вимог до майбутнього програмованого додатку:

- програмний застосунок містить працювати у операційних системах сімейства Windows версій 7-11;
- програмний застосунок повинен мати дружній, інтуїтивний призначений задля споживача вигляд;
- програмний застосунок містить споживати мінімум системних ресурсів задля свого функціювання;

Перелік вимог до функціонала програмованого додатку:

- спроможність вибору «клієнтського» та «зовнішнього» комп'ютерних систем із загального списку комп'ютерних систем, встановлених на РС;
- спроможність ручного введення фізичної адреси «зовнішнього» комп'ютерного пристрою;
- визначення та відображення на екрані адрес усіх комп'ютерних систем, котрі віщають у мережі;
- спроможність додавання та видалення дозволених комп'ютерних адрес (систем);
- облік потоку у кожному мережевому пристрою (у кожній MAC-адресі);
- спроможність запускати та зупиняти роботу додатку;
- спроможність виводити в режимі реальному часі результати

функціювання (дослідження та обробки потоку).

Таким чином, розроблюваний програмний продукт повинен працювати із необмеженою кількістю користувачів (клієнтів), потік з яких надходить на другий вигляд локального PC, перенаправляючи потік на перший вигляд, за яким спроможне слідувати роутер, підключений до мережі Інтернет чи інша локальна мережа. Приблизна узагальнена схема структури мережі, задля якої розробляється програмне середовища, виглядає наступним чином (рис.1.4).

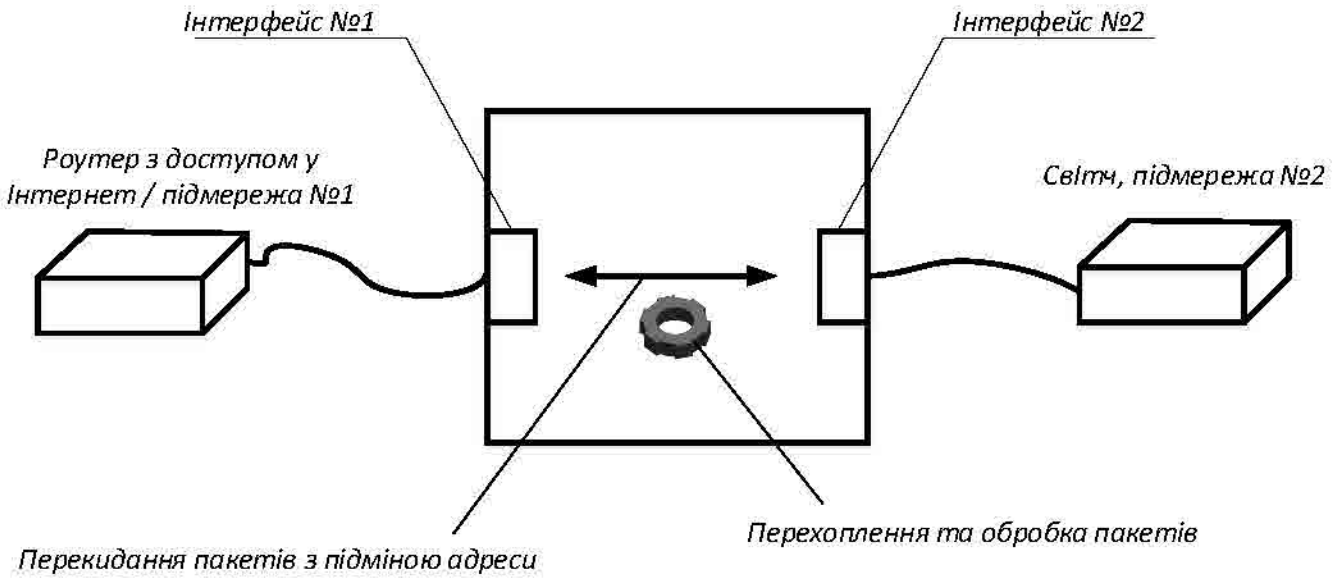



Рисунок 1.4. Узагальнена схема структури мережі задля розроблюваного ПЗ

## 1.5 Вибір ядра моделі комп'ютерного відстежування

Відповідно до проведеного огляду найкраще в якості ядра моделі відстежування мережі підходить структура Nagios, адже вона володіє такими якостями:

- є інструменти генерування діаграм;
- є інструменти генерування звітності;
- є спроможність логічного групування;
- існує вбудована структура запису трендів та тих прогнозування;
- є спроможність автоматичного додавання нових систем (Autodiscovery) за поміччю офіційного плагіна;
- є спроможність розширеного відстежування хоста із використанням

агента;

- підтримка правила SNMP через плагін;
- підтримка правила Syslog через плагін;
- підтримка зовнішніх скриптів;
- підтримка самописних плагінів та спроможність тих швидкого та простого створення;
- вбудовані тригери та події;
- повнофункціональний веб-вигляд;
- спроможність розподіленого відстежування;
- інвентаризація через плагін;
- спроможність зберігання інформації як у файлах, так та у базах інформації SQL, що дуже важливо при збільшенні обсягів;
- ліцензія GPL, але отже безкоштовна базова поставка, підтримка та відкриті вихідні коди ядра моделі та супроводжуючих компонентів;
- динамічні та налаштовувані карти;
- керування доступом;
- вбудована мова опису хостів, сервісів та перевірок;
- спроможність відстеження користувачів.

Структура мережного відстежування Zabbix володіє схожим набором параметрів, але на момент впровадження володіла набагато меншим функціоналом, ніж Nagios та мала статус beta-версії. Крім того, дослідження тематичних форумів та новинних стрічок показало найбільшу поширеність серед користувачів саме Nagios, що означає наявність написаної користувачами документації та максимально докладно описаних складних моментів в налаштуванні.

Nagios передбачає виконувати моніторинг таких комп'ютерних сервісів як SMTP, TELNET, SSH, HTTP, DNS, POP 3, IMAP, NNTP та багатьох інших. Крім такого, можливо стежити за використанням ресурсів серверів, таких як витрата дискового простору, вільна пам'ять та завантаженість процесора. Існує спроможність створювати свої власні обробники подій. Ці обробники

виконуватимуться при виникненні тих чи інших подій, ініційованих перевірками сервісів чи серверів. Такий підхід дозволить активно реагувати на події, що відбуваються та намагатися автоматично вирішувати виниклі задачі. Зокрема, можливо створити обробник подій, котрий буде самостійно перезапускати завислий сервіс. Ще однією перевагою моделі відстежування Nagios є спроможність керувати нею віддалено за поміччю wap-інтерфейса мобільного телефону. Ця якість, у свою чергу, допомагає відрізнити непрацюючі хости, з тих, що недоступні у даний момент через неполадки у роботі проміжних ланок. Nagios вміє будувати графіки функціювання спостережуваних систем та карти контрольованої комп'ютерної інфраструктури.

Багатьом із адміністраторів добре знайомий предок Nagios – NetSaint. Незважаючи на те, що сайт проекту NetSaint все ще справно функціонує, нові розроблювання базуються вже на вихідному коді Nagios. В документації, що поставляється із Nagios, йдеться, що він буде стабільно працювати та із багатьма іншими Unix-подібними системами. Задля відображення web-інтерфейса Nagios знадобиться сервер Apache. Можливо використовувати будь-котрий інший, але у даній роботі буде розглядатися саме Apache, як один із найбільш поширених.

## 1.6 Вибір бібліотек задля функціювання із мережевими пристроями і потіком

### 1.6.1 Бібліотека Pcap

Бібліотека Pcap (Packet Capture) передбачає створювати додатку визначення комп'ютерних інформації, що надходять на мережеву карту комп'ютера. Прикладом програмованого середовища, що використовує бібліотеку Pcap, служить програма Wireshark. Різноманітні додатку відстежування і перевірка мережі, сніфери, використовують цю бібліотеку. Вона призначена задля застосування спільно із мовами C/C++/C#, але задля функціювання із бібліотекою на інших мовах, таких як Java, .NET, використовують обгортки. Задля Unix-подібних систем є бібліотека libpcap, але задля Microsoft Windows – WinPcap. Програмне середовища комп'ютерного

відстежування спроможне використовувати libpcap чи WinPcap, щоб захопити пакети, котрі подорожують у мережі, та (у новіших версіях) задля передачі наборів в мережі. Libpcap та WinPcap разом з цим підтримують збереження захоплених наборів в файл та читання файлів, що містять збережені пакети.

Додатку, написані на основі libpcap чи WinPcap, спроможні захопити мережевий потік, аналізувати того. Іншими словами, задля того щоб перехоплювати пакети, мережева карта повинна працювати у так званому режимі promiscuous-mode. Задля функціювання у такому режимі потрібна підтримка на рівні драйверів, яку та забезпечує спеціальна бібліотеці WinPcap.

Склад WinPcap:

1) фільтр наборів на рівні ядра;

2) низькорівнева DLL (packet. dll);

3) високорівнева системно-незалежна бібліотека (wpcap. dll).

Єдиний недолік даної бібліотеки у тім, що вона працює не із усіма нестандартними адаптерами (Wi-Fi-картки, VPN та т.д.).

### 1.6.2 Бібліотеки SharpPcap та Packet. Net

SharpPcap – бібліотека задля .NET, котра передбачає перехоплювати пакети. У суті, є обгортка над бібліотекою Pcap, котра використовується у багатьох популярних продуктах, зокрема, сніфер Wireshark, IDS Snort. ІЗ SharpPcap разом з цим поставляється чудова бібліотека задля парсингу наборів – Packet. Net. Packet. Net підтримує наступні протоколи: Ethernet, LinuxSLL, Ip (IPv4 and IPv6), Tcp, Udp, ARP, ICMPv4 та ICMPv6, IGMPv2, PPPoE, PTP, Link Layer Discovery Protocol (LLDP), Wake-On-LAN (WOL).

## 1.7 Вибір допоміжних комп'ютерних систем

### 1.7.1 Віртуальна машина Vm-ware

Vm-ware Workstation – програмне середовища віртуалізації, призначене задля комп'ютерів x86-64 операційних систем Microsoft Windows та Linux. Передбачає користувачу встановити одну чи більше уявних машин на один

фізичний комп'ютер та запускати тих паралельно із ним. Кожна віртуальна машина спроможне виконувати свою операційну систему, включаючи Microsoft Windows, Linux, BSD, та MS-DOS. Vm-ware Workstation розроблена компанією Vm-ware, підрозділом EMC Corporation. Vm-ware Workstation підтримує мости із мережним адаптером реального комп'ютера, але разом з цим створення спільних папок із віртуальною машиною. Програма спроможне монтувати ISO-образи в віртуальні оптичні приводи, при цьому віртуальна машина буде вважати, що приводи справжні. Віртуальні жорсткі диски зберігаються в файлах .vmdk. Vm-ware Workstation у будь-котрий момент спроможне зберегти поточний стан віртуальної машини (знімок). Дані знімки пізніше спроможні знаходитись відновлені, що повертає віртуальну машину в збережений стан. Vm-ware Workstation включає у себе спроможність об'єднувати декілька уявних машин у групу, яку можливо включати, вимикати, призупиняти чи відновлювати як єдиний об'єкт, що є корисним задля перевірка технологій клієнт-сервер. Віртуальні машини на платформі Vm-ware дозволяють користувачам створювати різні комбінації уявних систем, що працюють за різними принципами комп'ютерної взаємодії.

Основою мережі Vm-ware є такі компоненти:

- віртуальні комутатори (Virtual Switches);
- віртуальні мережеві інтерфейси (Virtual Ethernet Adapters);
- віртуальний міст (Virtual Bridge);
- вбудований DHCP-сервер;
- пристрій трансляції комп'ютерних адрес (NAT, Network Address Translation).

Фундаментальним елементом комп'ютерної взаємодії у Vm-ware Workstation є віртуальний комутатор. Він забезпечує мережеву взаємодію уявних машин на манер фізичного пристрою: на віртуальному комутаторі є порти, до яких спроможні знаходитись прив'язані віртуальні мережеві інтерфейси уявних машин, але разом з цим інші компоненти віртуальної інфраструктури у межах хоста. Вбудований DHCP-сервер Vm-ware передбачає віртуальним машинам

автоматично отримувати IP-адресу в своїй підмережі, але віртуальний NAT-пристрій забезпечує трансляцію комп'ютерних адрес при спілкуванні уявних машин із зовнішньою системою. Продукти Vm-ware Workstation надають користувачам спроможність призначити віртуальній машині один із трьох базових типів комп'ютерної взаємодії задля кожного із уявних комп'ютерних адаптерів:

- Bridged;
- Host-only;
- NAT.

Bridged Networking – тип комп'ютерної взаємодії, котрий передбачає прив'язати мережевий адаптер віртуальної машини до фізичного комп'ютерного інтерфейса комп'ютера, що дає спроможність розділяти ресурси комп'ютерної карти поміж хостовою та віртуальною системою. Віртуальна машина із таким типом комп'ютерної взаємодії буде вести себе у відношенню до зовнішньої мережі хостової моделі як незалежний комп'ютер. Структура Bridged Networking наведена на рис. 1.5.

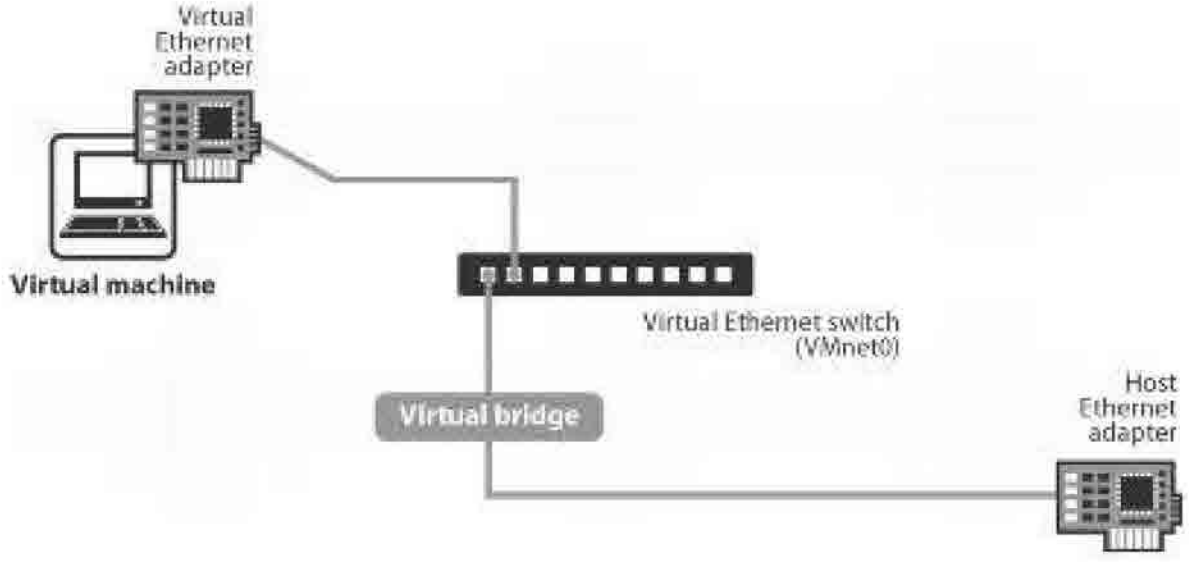



Рисунок 1.5. Структура комп'ютерної взаємодії у режимі Bridged Networking

Host-Only Networking – тип комп'ютерної взаємодії, оптимальний задля цілей перевірка програмованого середовища, коли потрібно організувати віртуальну мережу у межах хоста, але віртуальним машинам не потрібний вихід у зовнішню мережу. Структура Host-Only Networking наведена на рис. 1.6.

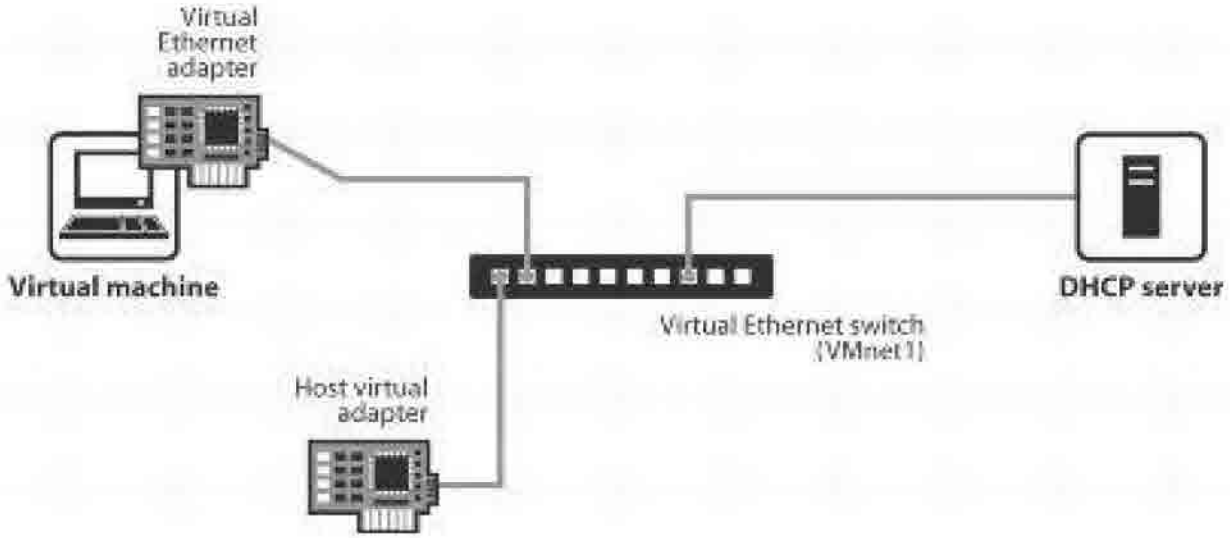



Рисунок 1.6. Структура Host-Only Networking

NAT Networking – цей тип комп'ютерної взаємодії дуже схожий на Host-Only, за одним винятком: до віртуального комутатора VMnet8 підключається пристрій трансляції IP-адрес (NAT). До такого комутатора разом з цим підключається DHCP-сервер, що роздає віртуальним машинам адреси із заданого діапазону (за замовчуванням I92.I68.89.128 – I92.I68.89.254) та, безпосередньо, самі віртуальні машини. NAT-пристрій передбачає здійснювати трансляцію IP-адрес, що передбачає віртуальним машинам ініціювати із'єднання у зовнішню мережу, не надаючи при цьому механізму доступа до уявних машин ззовні. Структура NAT Networking наведена на рис. 1.7.

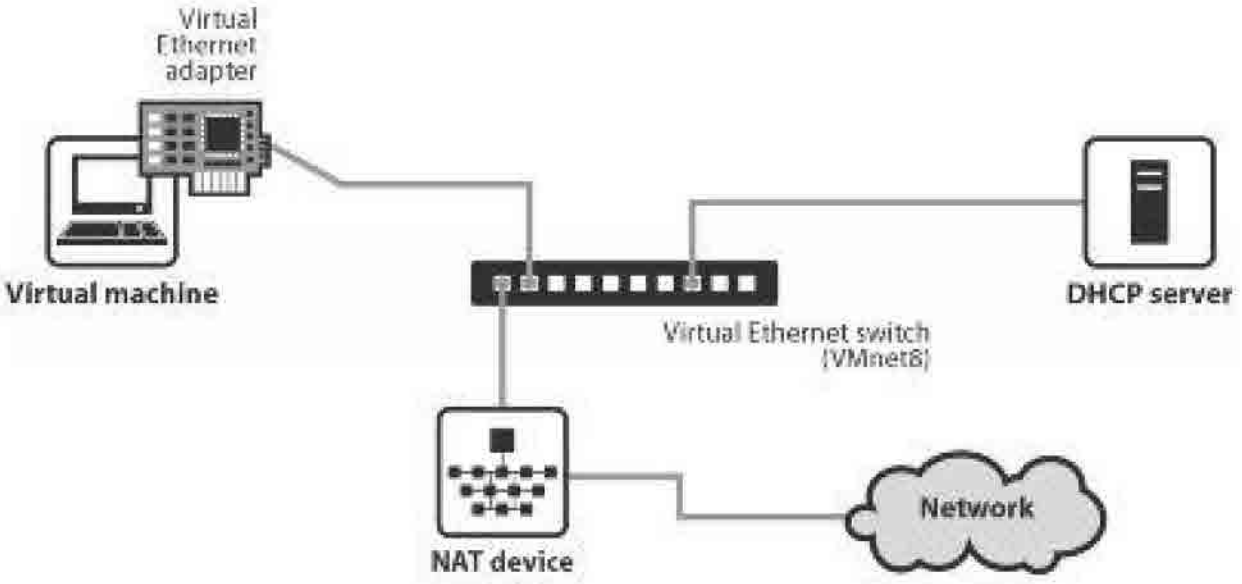



Рисунок 1.7. Структура NAT Networking

У даному випадку, при створенні та налаштуванні стенду, будемо використовувати підключення Host-Only.

### 1.7.2 Визначник потоку Wireshark

Wireshark – визначник комп'ютерних комутацій Ethernet та потоку всередині тих. Wireshark надає функціонал схожий із додатком tcpdump, проте на відміну з нього містить графічний вигляд та надає більше можливостей у фільтрації та подальшого сортування інформації. Програма переглядає весь потік, що проходить у мережі у режимі реального терміну. Досягається є перекладом комп'ютерних карт у нерозбірливий режим (promiscuous mode).

Wireshark спроможне розібрати будь-котрий мережевий пакет та відобразити числа усіх того полів. Досягається є завдяки знанню структур найрізноманітніших комп'ютерних правил. Застосування бібліотеки pcap дає спроможність захопити дані лише із тих комутацій, котрі підтримуються цією бібліотекою, але Wireshark уміє працювати та із безліччю інших форматів вхідних інформації. Є дає спроможність розширити можливості захоплення відкриттям файлів інформації, захоплених іншими програмами.

## 1.8 Створення і налагодження стенду задля перевірка ПЗ

Задля налагодження та перевірка розроблюваних комп'ютерних систем діагностування сегментів обчислювальної мережі треба створити та налаштувати стенд. Задля створення внутрішньої мережі із трьох персональних комп'ютерів використано продукт Vm-ware workstation. В ролі двох клієнтських PC прийнято рішення використовувати PC на базі операційних систем Windows IO. Орім такого, на PC були встановлені:

- Інтегроване середовище розроблювання Embarcadero RAD Studio IO.3;
- визначник наборів Wireshark із бібліотекою WinPcap;
- бібліотеки SharpPcap. dll та Pcap. NET. dll.

Першій клієнтській машині був призначений IP: I92.I68.115.IO, шлюз (Gateway): I92.I68.115.20, MAC-адреса: 00:0C:29:62:89:AB. Другій машині

призначені IP: I92.I68.116.IO, шлюз: I92.I68.116.20, MAC-адреса: 00:0C:29:B3:66:CA. Основна машина (із двома інтерфейсами) разом з цим отримала мережеві адреси.

Задля першого комп'ютерного адаптера:

- IP: I92.I68.115.IO;
- MAC: 00:0C:29:17:5E:3C;

Задля другого комп'ютерного адаптера:

- IP: I92.I68.116.IO;
- MAC: 00:0C:29:17:5E:46.

Підключивши першу машину до першого мережного інтерфейса головної машини, але другу машину до другого інтерфейса, отримали стенд із мінімальною кількістю систем, необхідних задля розроблювання та перевірка програмованого середовища. Схема підключення представлена на рис. 1.8. Після підключення зв'язок поміж машинами був перевірений командою Пінг.

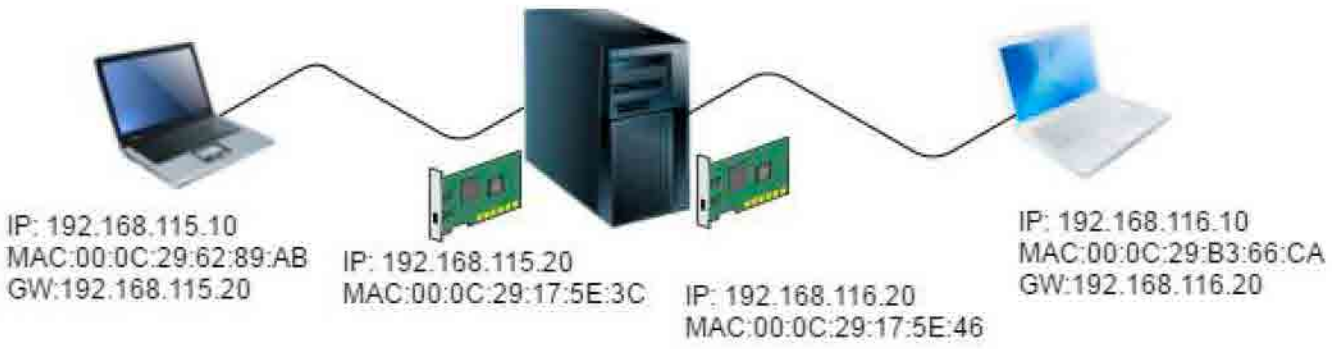



Рисунок 1.8. Схема стенду задля перевірка розроблюваного ПЗ

## 1.9 Підготовка комп'ютерних інструментів розроблювання

При роботі в локальних чи глобальних обчислювальних мережах в деяких випадках треба знати поточний стан як локального, так та віддалених хостів (чи містить локальний хост в цей момент спроможність виходу у мережу Інтернет, чи наявний якийсь віддалений хост та т.д.). Так, задля реалізації поставлених цілей при написанні додатку буде використовуватися мова програмування Delphi і інструменти розроблювання Embarcadero RAD Studio IO.3. Необхідні параметри мережі програма отримує за поміччю інтерфейса I-C-

M-P API. Принцип функціювання пінг заснований на використанні правила I-C-M-P – Internet Control Message Protocol (протокол керуючих чи контрольних повідомлень). За поміччю I-C-M-P хости в мережі обмінюються різною службовою інформацією (інформацією щодо зміну маршруту, зменшенні швидкості передачі, відсутність доступа до якої-небудь адреси та т.д.) У основі правила I-C-M-P лежить поняття повідомлень. Повідомлення I-C-M-P-правила, як правило, сповіщають щодо помилки, що виникають при обробці датаграм. I-C-M-P використовує основні властивості правила IP так, якби він був протоколом більше високого рівня. Насправді ж I-C-M-P є складовою частиною IP.

Одним із типів повідомлень правила є так званий тип повідомлень "ехо-запит". Одержавши повідомлення "ехо-запит", хост зобов'язаний відповісти повідомленням типу "ехо-відповідь". У суті, "ехо-запит" й "ехо-відповідь" відрізняються лише адресами відправника та одержувача і кодом типу повідомлення (тип 8 – "ехо-запит", тип 0 – "ехо-відповідь"). Більше докладно протокол I-C-M-P описаний у RFC792. Реалізації утиліти пінг на різних платформах істотно відрізняються. Так, у ОС UNIX використаються RAW sockets (неопрацьовані, "сирі" сокети), але в ОС Windows усіх версій – так званий I-C-M-P API. Застосування I-C-M-P API виправдане лише у ОС Windows, тім що задля застосування RAW sockets в цих ОС необхідні права Адміністратора.

## 1.10 Розробка блок-схем алгоритмів і тих реалізація

Треба зазначити, що задля діагностування сегментів обчислювальної мережі (зокрема, мережі ISP – Internet Service Provider чи просто масштабних корпоративних локальних ресурсів) все гостріше відчувається нестача універсального автоматизованого алгоритму пошуку і усунення несправностей, що зменшує трудовитрати системних адміністраторів на обслуговування та виявлення / усунення неполадок. Опорною точкою задля реалізації програмованого середовища діагностування сегментів обчислювальної мережі за

протоколом I-C-M-P стане узагальнена блок-схема алгоритму пошуку і усунення несправностей (траблшутінга) [1], котра зображена на рис. 1.9.

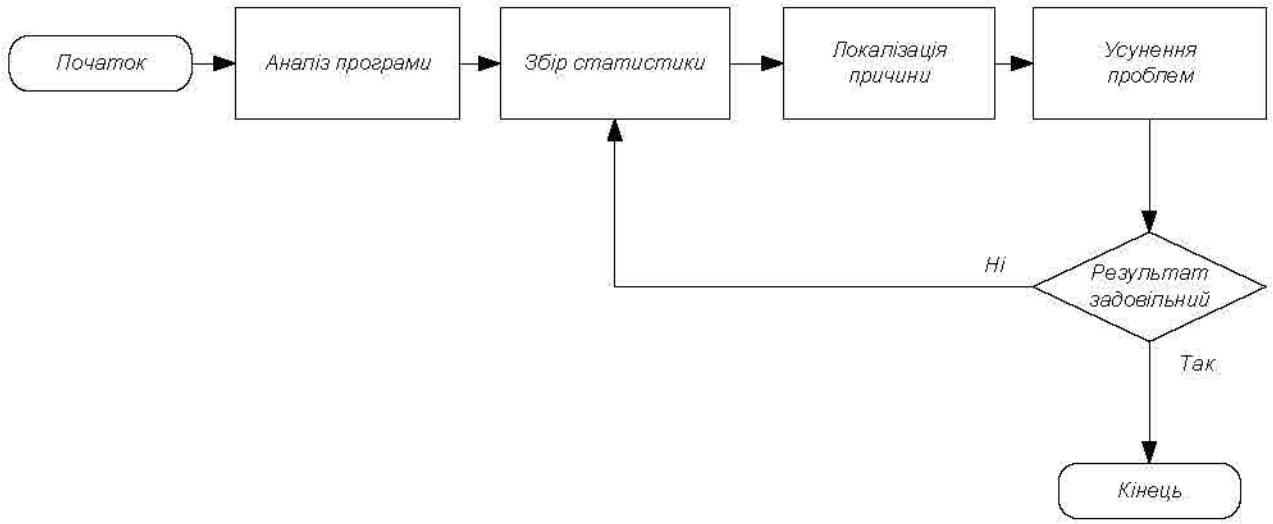



Рисунок 1.9. Узагальнений алгоритм пошуку несправностей в мережі

Маючи певну структуру, можливо, відштовхуючись з кожної складової блоку, сформувати реальні послідовності дій, виконання яких приведе до вирішення будь-якої позаштатної ситуації (чим докладнішим та розгалуженішим буде алгоритм, тим більше гарантій вирішення саме будь-якої позаштатної ситуації). Разом з цим треба відзначити, що задля розглянутих вище комутацій, у яких буде використовуватися такий алгоритм, при виникненні будь-якого масштабу несправності відлік терміну на її усунення йде на хвилини. Задля інтернет-провайдера є заявлений період простою, аптайму устаткування. Задля корпоративних комутацій є, у першу чергу, збитки, котрі обчислюються часом навіть мільйонами через зірвані угоди чи втраченого великого партнера/клієнта. Зрозуміло, задля таких ситуацій передбачаються резервні канали, устаткування і цілі гілки маршрутів, але ніхто не застрахований з випадковостей.

Таким чином, що розроблюваний алгоритм повинен ефективно працювати як на великих мережах із великою кількістю сегментів, максимально швидко виконуючи свої процедури, так та на більше дрібних комп'ютерних структурах, задля яких можливо разом з цим за підсумками визначення надавати деякі рекомендації із розвитку і резервування.

### 1.10.1 Аналіз задачі в мережі, збір комп'ютерної статистики

Зважаючи на обумовлені вище умови швидкодії алгоритму, пропонується об'єднати перші два етапи початкової моделі, щоб ефективно та у найкоротші терміни провести всі необхідні опитування та перейти до усунення задачі, але у іншому випадку видати список рекомендацій до дії обслуговуючого персоналу.

Даний етап повинен включати у себе наступні процеси:

- Первинне опитування кожного вузла мережі на предмет встановлення факту фізичної доступності. Є означає, що із початкового кореневого вузла будуть опитані у черзі кожен елемент мережі за поміччю команди пінг;
- Коли буде виявлено, що пристрій не відповідає, того доступність треба перевірити шляхом спроби зайти на нього (telnet, ssh) разом із опитуванням сусідніх із ним систем на предмет доступності інтерфейсів несправного пристрою;
- Коли спроби зайти на проблемний вузол мережі виявилися безуспішними, але опитування інтерфейсів пристрою із сусідніх йому сегментів показує тих недоступність, із великою ймовірності цей вузол мережі вийшов із ладу та на ньому як мінімум відсутнє живлення. Є означає, що ремонт та діагностику треба проводити фізично на місці, завершуючи роботу алгоритму [2].

Алгоритм задавання початкових параметрів задля розроблюваного ПЗ діагностування сегментів обчислювальної мережі показаний на рис. 1.IO.

### 1.10.2 Локалізація причини несправності в мережі

Коли всі пристрої доступні, тих інтерфейси функціонують, але робота мережі здійснюється некоректно, є означає задачі чи із маршрутизацією, чи зі зміною одного чи декількох сегментів мережі. Звідси випливає необхідність локалізувати несправну ділянку та із'ясувати ключовий пристрій, після взаємодії із яким механізм транспортування потоку порушується. У такому випадку треба спиратися на факти наслідків несправності, за якими можливо визначити проблемну ділянку.

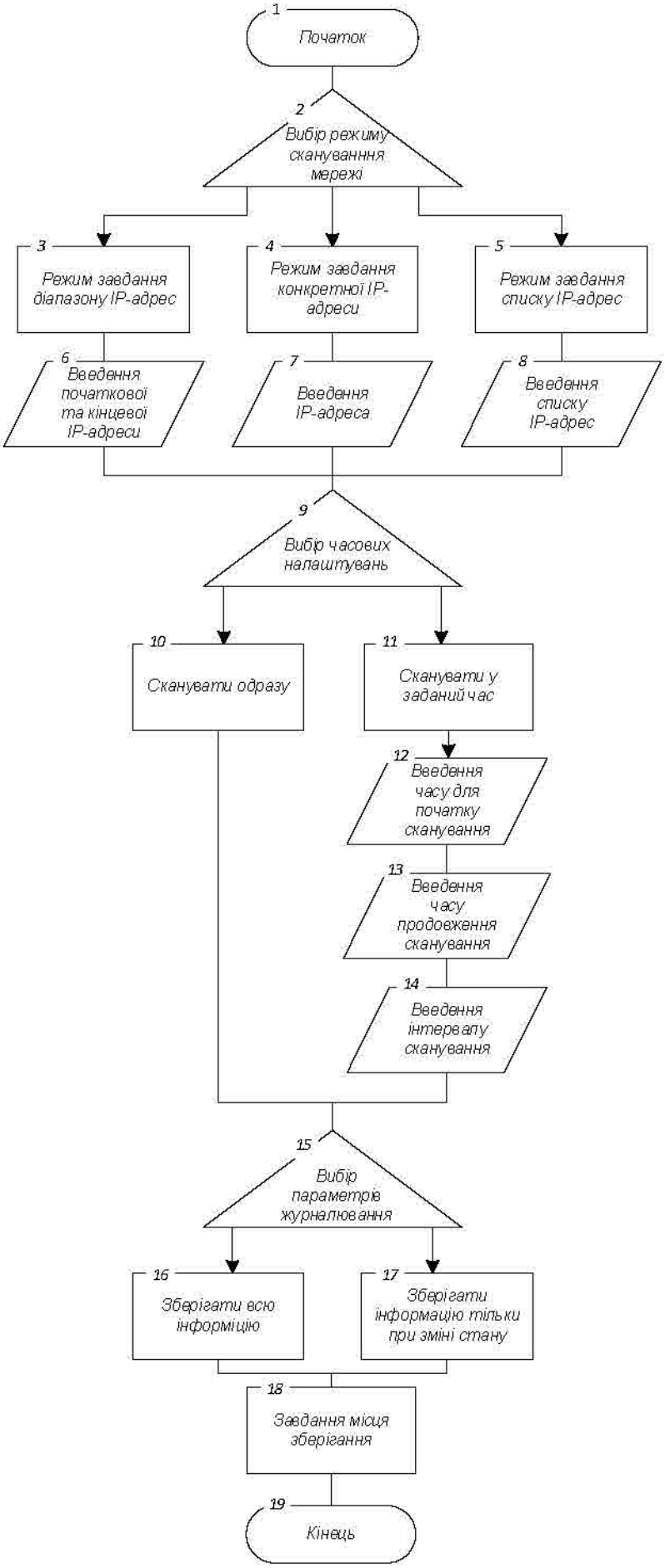



Рисунок 1.10. Алгоритм задавання початкових параметрів

Є спроможне означати як скарги користувачів певної підмережі, так та результат спостережень мережі за поміччю утиліт відстежування, котрі достовірно методом SNMP-опитування покажуть несправні вузли.

Відносно ж автоматизації процесу, припустимим є застосування команди traceroute, котра буде як сканер запущена алгоритмом послідовно із різних сегментів мережі до найбільш віддалених, тим самим буде із'ясована ділянка мережі, на якій виникає непрохідність потоку [3].

### 1.10.3 Усунення задачі в мережі і відновлення працездатності

Локалізувавши, таким чином, несправну ділянку, хід переходить до наступного етапу вирішення виниклої несправності, але саме – її усунення. Є найбільш великий блок алгоритму, котрий включає у себе наступні пункти:

- Один із інтерфейсів спроможне знаходитись відключений логічно, тобто адміністративно перебувати в стані down. Перевірка здійснюється командою run show interfaces terse. Числа параметрів Admin та Link, що вказують на фізичну та адміністративну доступність інтерфейсів, містить знаходитись Up. У іншому випадку, коли числа параметра Admin = Down, відповідно вигляд із якихось причин відключений адміністративно та того роботу треба відновити командою delete interfaces <InterfaceName> disable (виконується із конфігураційного режиму). Коли ж параметр Link = Down, то у такому випадку мова йде щодо фізичну несправність каналу зв'язку. Видається відповідне інформаційне повідомлення;
- Коли апаратно маршрутизуючий пристрій працює справно, є означає, що порушення функціювання стосується лише логіки функціювання того програмної частини – конфігурації. У такому випадку перевірці підлягає саме конфігурація пристрою, котра, як правило, є типовою у великих мережах. Є означає, що із невеликими відмінностями у адресах та логічних зв'язках структура конфігурації кожного маршрутизатора фактично однакова, тім що реалізована схема маршрутизації

забезпечується функціонуванням відповідних правил, налаштування яких у кожному пристрої є регламентованою процедурою. Іншими словами, у першу чергу за шаблоном перевіряються налаштування ключових розділів конфігурації маршрутизаторів – protocols, routing – options, policy – options, interfaces, де все містить знаходитись приведене до стандартного шаблону, але адресація буде вірною [4].

Таким чином, початковий алгоритм (рис.1.9) набуває розширеного вигляду (рис.1.11), де:

- Блок 1 – процедура опитування кожного вузла мережі командою пінг;
- Блок 2 – процедура перевірки маршрутів мережі поміж крайніми точками командою traceroute;
- Блок 3,5 – умова-перевірка недосяжних / недоступних систем;
- Блок 4,6 – умова-перевірка виходу із циклу дослідження;
- Блоки 1,3,4 та 2,5,6 об'єднані у умовну процедуру перевірки вилученої доступності сегментів мережі пінг та traceroute відповідно;
- Блок 7 – процедура авторизації на потенційно нефункціонуючих пристроях;
- Блок 8 – умова-перевірка успішності авторизації;
- Блок 11 – висновок повідомлення за умови віддаленої недоступності пристрою;
- Блок 9 – процедура перевірки працездатності інтерфейсів;
- Блок 12 – умова-перевірка функціювання усіх інтерфейсів;
- Блок 13 – умова-перевірка виду недоступності інтерфейса (адміністративно чи фізично);
- Блок 10 – активація інтерфейса в разі адміністративного відключення інтерфейса;
- Блок 14 – виведення повідомлення щодо фізичні проблемі каналу зв'язку даного інтерфейса;

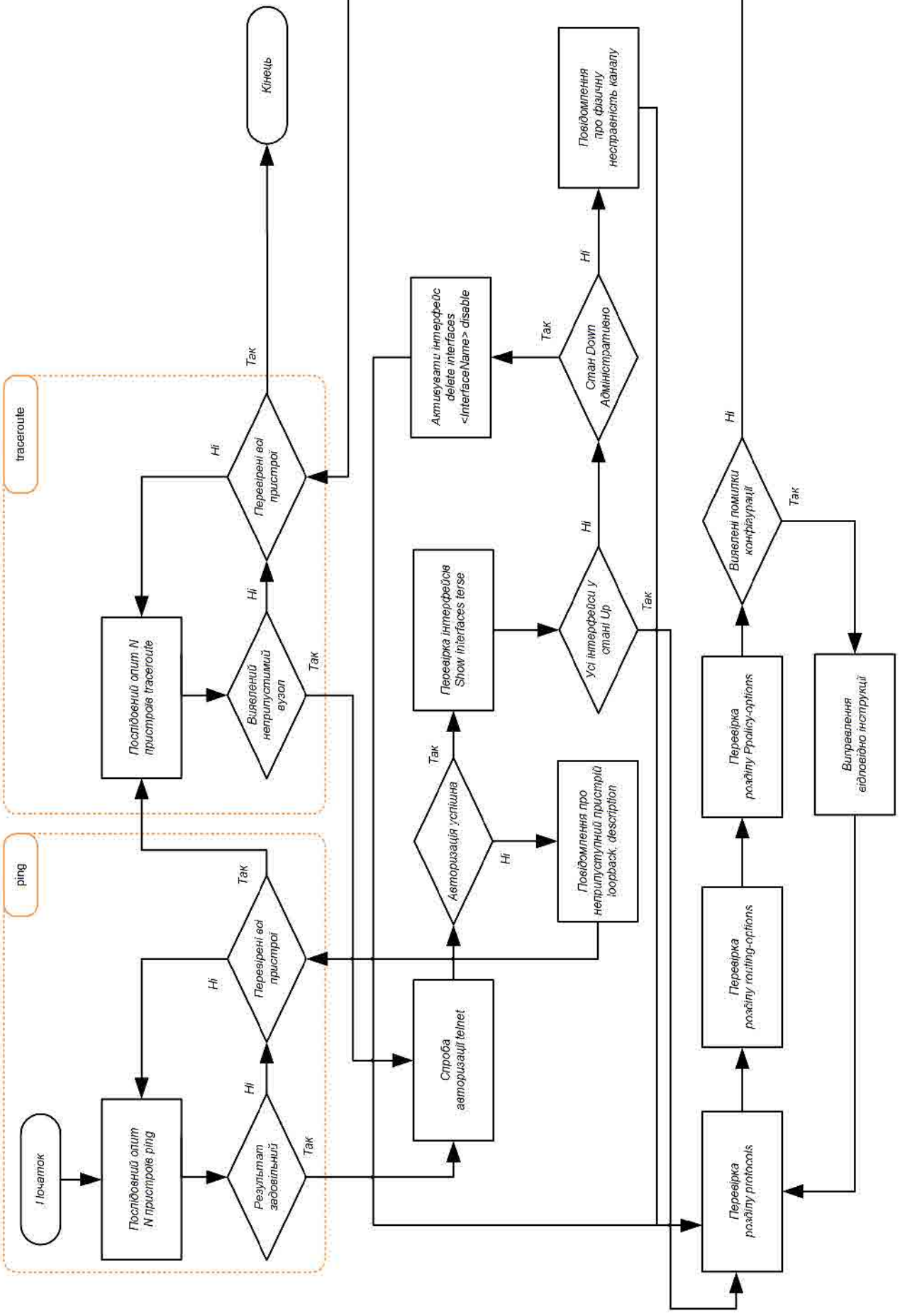



Рисунок 1.11. Розширений алгоритм діагностування мережі

- Блок 15 – перевірка розділу конфігурації protocols шляхом порівняння зі зразком та перевірки правильності налаштування логіки функціювання;
- Блок 16 – перевірка розділу конфігурації routing-options шляхом зіставлення зі зразком, виявлення помилкових маршрутів;
- Блок 17 – перевірка розділу конфігурації policy-options шляхом зіставлення зі зразком, перевірки правильності функціювання логіки налаштованих фільтрів;
- Блок 18 – умова-перевірка наявності помилок конфігурації у вищезазначених розділах;
- Блок 19 – процедура внесення змін в відповідності із виявленими невідповідностями.

### 1.10.4 Реалізація команди Пінг

Команда пінг служить задля примусового виклику відповіді конкретної машини. Задля такого використовується дейтаграма ECHO_REQUEST правила I-C-M-P. Є протокол низького рівня, що не вимагає наявності серверних процесів на зондуючій машині; є гарним способом переконатися у тім, що напруга є та IP перебуває в робочому стані. Успішний результат застосування команди пінг зовсім не обов'язково означає, що виконуються котрі-небудь сервісні додатку високого рівня.

Пінг – добрий засіб перевірки правильності конфігурації мережі, оскільки в виконанні цієї команди беруть участь структура маршрутизації, схеми дозволу адрес та мережні шлюзи. Коли дана команда не працює – більше складні інструменти тим більше не функціонують. Незважаючи на свою простоту, пінг – один із основних елементів, що використовують при налагодженні комутацій.

Задля практичної реалізації, як завжди, можливо використати кілька підходів. Перший із тих – застосування низькорівневих функцій API (зокрема, вбудованих в бібліотеку I-C-M-P.DLL). Другий – застосування компонентів високого рівня (зокрема, Indy IdICMPClient).

ТА у першому та у другому способі є свої позитивні та негативні моменти.

Так, при використанні функцій API відкомпільований код буде мати набагато менші розміри, ніж при використанні компонентів високого рівня, але продуктивність того буде вище (зокрема, при одночасному пінгуванні однієї підмережі із використанням потоків).

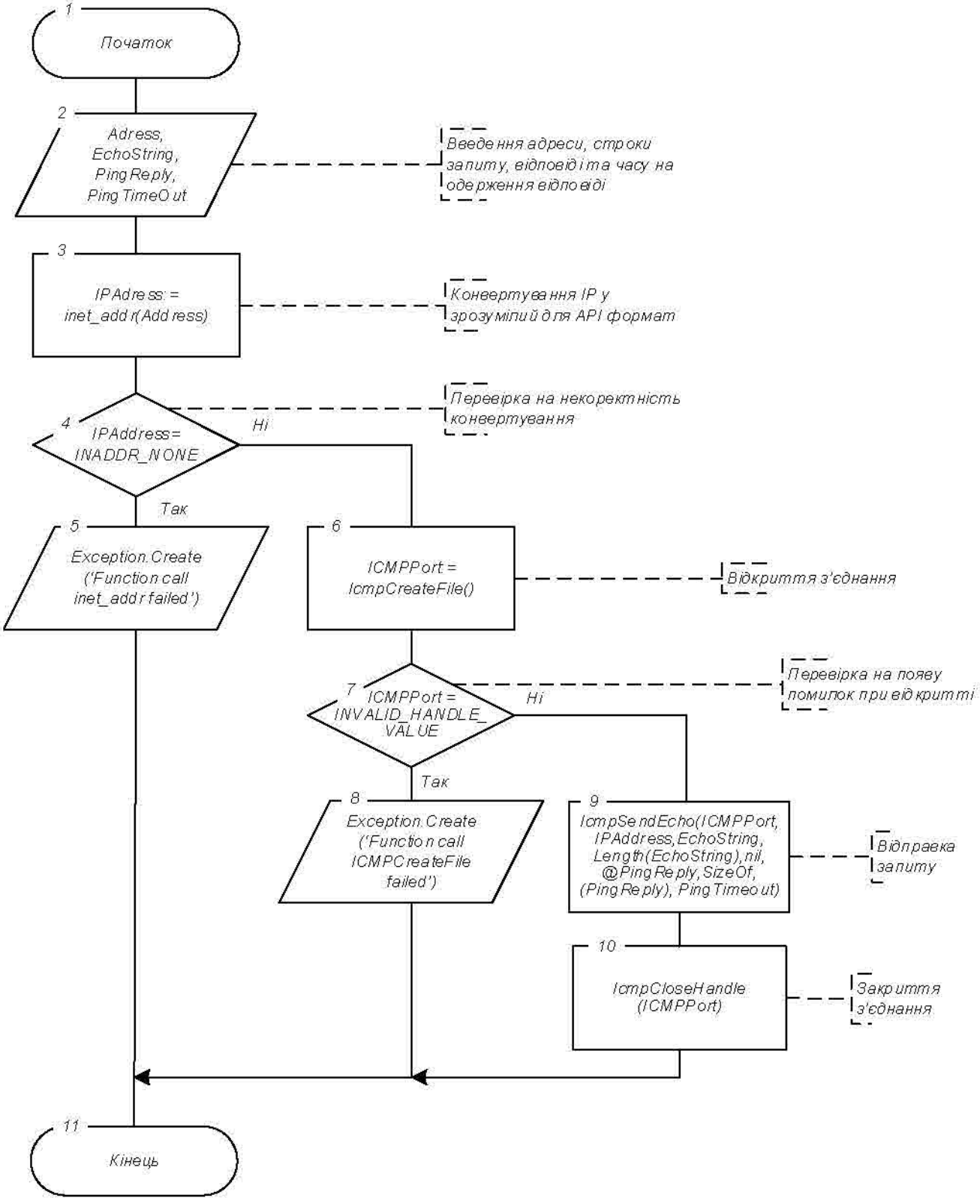



Рисунок 1.12. Алгоритм функціювання створеної процедури Пінг

ІЗ іншого боку, компоненти можливо використати, маючи лише віддалені поняття щодо роботу із протоколом I-C-M-P, але разом з цим щодо застосування Windows API. Але, в той же період, компоненти породжують невиправдано великий код задля виконання, що призводить до пониження продуктивності в цьому випадку. На сьогодні розробниками процесорів та пам'яті майже стерта грань поміж продуктивністю коду, написаного із використанням Windows API, та коду, написаного із використанням систем більше високого рівня абстракції (в нашому випадку – компонентів ICP Embarcadero RAD Studio IO.3 Delphi). При розробці додатку буде використане пінгування, написане як із використанням WinAPI так та із використанням компонентів Delphi Indy (рис.1.12).

### 1.10.5 Windows API

При використанні Windows API задля написання процедури пінгу використано бібліотеку I-C-M-P (i-c-m-p.dll), що надає вигляд задля функціювання із однойменним протоколом. У інтерпретації Delphi тих оголошення виглядають як показано на рис.1.13. Перша із тих – IcmpCreateFile створює із'єднання, із яким ми збираємося працювати. Друга функція – IcmpCloseHandle закриває із'єднання, із яким ми працювали. За поміччю третьої процедури IcmpSendEcho відбувається посилання через установлене із'єднання відповідних інформації.

```
function IcmpCreateFile:Thandle; StdCall;

function IcmpCloseHandle (H:Thandle):Bool; StdCall;

function IcmpSendEcho (IcmpHandle:Thandle;
DestinationAddress:TipAddr;
RequestData:pointer;
RequestSize:word;
RequestOptions:POption_Information;
ReplyBuffer:pointer;
ReplySize:integer;
Timeout:integer):Integer; stdcall;
```

Рисунок 1.13. Бібліотека I-C-M-P

Зупинимося докладніше на процедури IcmpSendEcho. Прийняті нею параметри використовуються як:

IcmpHandle – ідентифікатор із'єднання, установленого за поміччю IcmpCreateFile;

DestinationAddress – адреса пінгуємого хосту;

RequestData – буфер із даними, котрі посилають при запиті;

RequestSize – розмір буфера запиту;

RequestOptions – додаткові властивості запиту;

ReplyBuffer – адреса буфера задля прийому результату;

ReplySize – розмір буфера прийому;

Timeout – період, протягом якого ми очікуємо відповіді з хосту.

Результатом процедури є кількість записів типу I-C-M-P_ECHO_REPLY, збережених у ReplyBuffer. Статус кожного запису зберігається в відповідному полі такого запису. При невдалому виклику функція повертає числа NULL; додаткова інформація доступна при виклику GetLastError.

Структура запису I-C-M-P_ECHO_REPLY відображена на рис 1.14:

```
Ticmp_echo_reply=record
Address: TipAddr; // Адреса, що відповіла
Status: integer; // Статус відповіді
RoundTripTime: integer; // Час проходження пакету
DataSize: word; // Розмір даних відповіді в байтах
Reserved: word; // Зарезервовано
Data: pointer; // Вказівник на буфер з відповіддю
Options: Toption_Information; // Опції відповіді
End;
```

Рисунок 1.14. Запис i-c-m-p_echo_reply

Разом з цим іноді зручніше використовувати розширений варіант структури i-c-m-p_echo_reply, котрий зображено на рис 1.15:

```
TsmICMP_Echo_Reply=record
Address: TipAddr; // Адреса, що відповіла
Status: integer; // Статус відповіді
RoundTripTime: integer; // Час проходження пакету
DataSize: word; // Розмір даних відповіді в байтах
Reserved: word; // Зарезервовано
Data: pointer; // Вказівник на буфер з відповіддю
Options: Toption_Information; // Опції відповіді
Data: array [0..255] of Char;
end;
```

Рисунок 1.15. Розширений варіант структури i-c-m-p_echo_reply

Зараз задля реалізації пінга хосту треба виконати наступні дії:

- створити із'єднання поміж хостами;
- викликати функцію icmpsendecho;
- обробити отриманий результат функціювання процедури icmpsendecho;
- закриваємо із'єднання поміж хостами.

Ці дії зручно оформити в вигляді процедури, котра приведена на рис. 1.16.

```
procedure Ping (const Address, EchoString: PChar;
var PingReply: TsmICMP_Echo_Reply;
const PingTimeout: Integer = 500);
var
IPAddress: TipAddr;
ICMPPort: THandle;
begin
// Конвертація IP в зрозумілий для API формат
IPAddress:= inet_addr (Address);
// Перевірка коректності конвертації
if (IPAddress = INADDR_NONE) then
begin
raise Exception.Create ('Function call inet_addr failed. ' +
'The IP address is probably invalid.');
end;
// Відкриті з'єднання
ICMPPort:= IcmpCreateFile ();
// Перевірка правильності відкриття
if (ICMPPort = INVALID_HANDLE_VALUE) then
begin
raise Exception.Create ('Function call IcmpCreateFile failed.');
end;
// Відправлення запиту "пінг"
IcmpSendEcho (ICMPPort, IPAddress,
EchoString, Length (EchoString), nil,
@PingReply, SizeOf (PingReply), PingTimeout);
// закриття з'єднання
IcmpCloseHandle (ICMPPort);
end;
```

Рисунок 1.16. Процедура пінг

Тепер при використанні у коді додатку конструкції, котра наведена на рис. 1.17, в змінній Reply ми одержимо результат пінгу.

```
Ping ('127.0.0.1',nil,Reply,5000);
```

Рисунок 1.17. Виклик команди пінг

### 1.10.6 Застосування Indy

При розробці додатку іноді треба працювати із убудованими у Delphi компонентами задля функціювання із системою. Зокрема, задля реалізації пінгу на рівні об'єктно-орієнтованого програмування неодноразово використовується компонент IdICMPClient зі складу Indy. Задля такого треба створити спочатку на порожній формі екземпляр класу TIdICMPClient, перетягнувши того із палітри компонентів Indy Clients, потім помістити на форму стандартну кнопку (TButton) та у її реакції на натискання мишкою записати код, котрий показано на рис. 1.18.

```
Self.IdIcmpClient1.Host:='localhost';//вместо 127.0.0.1 здесь можно
использовать имя "localhost"
Self.IdIcmpClient1.Ping;
```

Рисунок 1.18. Код задля функціювання екземпляру класу TIdICMPClient

Разом з цим задля нормальної функціювання треба виставити відповідний інтервал очікування відповіді, у завершенні якого, чи при випадку одержання інформації з пінгуємого абонента, буде викликатися оброблювач, зображений на рис. 1.19.

```
TidIcmpClient.OnReply (Sender:TComponent; const
AReplyStatus:TReplyStatus);
```

Рисунок 1.19. Оброблювач відповідей

Оброблювач містить реалізацію виведення відповідних інформації, що були отримані при використанні написаної процедури пінгування і відображення на екрані задля інформування адміністратора чи іншого додатку (рис. 1.20).

```
procedure TForm1.IdIcmpClient1Reply (ASender: TComponent;
const AReplyStatus: TReplyStatus);
begin
ListBox1.Items.Add ('Reply:'+IntToStr
(AReplyStatus.MsRoundTripTime));
end;
```

Рисунок 1.20. Реалізація інформації пінгу

Звичайно, задля пінгу віддаленої машини ми завжди можемо звернутися до відповідної додатку командного рядка, оскільки 100% операційних систем

підтримують цю команду, але не завжди справа обмежується одним пінгом – особливо коли ми реалізуємо власний вигляд поверх IP. В такому випадку виклик зовнішньої додатку при кожній перевірці неприпустимий.

### 1.10.7 Структурні елементи інтерфейса програмованого додатку

В структурі інтерфейса створюваного додатку задля діагностування сегментів обчислювальної мережі передбачені:

- демонстраційні елементи;
- елементи керування.

Демонстраційні елементи містять:

- поля, що відображають клас внутрішньої обчислювальної мережі;
- поля, що відображають діапазон IP-адрес задля дослідження;
- поле, що відображає поточну скановану IP-адресу абонента в мережі;
- поле, що відображає список IP-адрес споживачів в мережі задля дослідження;
- поле, що відображає, у якому файлі виконується запис процесу дослідження внутрішньої обчислювальної мережі;
- поле, в якому зазначено, коли розпочнеться хід дослідження;
- поле, в якому зазначено, коли закінчиться хід дослідження;
- поле, в якому зазначений інтервал поміж процедурами дослідження;
- таблиця, у якій відображена інформація щодо стани сканованих споживачів внутрішньої обчислювальної мережі в заданий період;
- поле із індикацією кількості недоступних споживачів;
- поле із індикацією IP-адрес, доступних в заданий період.

Елементи керування включають в своєму складі:

- поля задля задавання класу внутрішньої обчислювальної мережі;
- поля задля задавання діапазону IP-адрес задля дослідження;
- поле, задля задавання файлу у якому буде виконуватися запис процесу дослідження внутрішньої обчислювальної мережі;
- набір задавання режимів відстежування внутрішньої обчислювальної

мережі;

- набір задавання дій задля виконання;
- набір задавання параметрів задля терміну;
- набір задавання режимів запис;
- поля задля задавання терміну задля початку дослідження;
- поля задля задавання інтервалу терміну поміж скануваннями;
- кнопки прийому інформації в демонстраційні поля.

## 1.11 Реалізація функціоналу розроблюваного додатку

Функціонально розроблювані програмні інструменти діагностування сегментів обчислювальної мережі за протоколом I-C-M-P забезпечують:

- задавання класу внутрішньої обчислювальної мережі;
- задавання діапазону IP-адрес споживачів мережі задля тих дослідження;
- задавання терміну, із якого розпочнеться дослідження споживачів мережі;
- задавання режимів відстежування внутрішньої обчислювальної мережі;
- обчислення та індикацію кількості доступних споживачів;
- підрахунок та індикацію кількості недоступних споживачів;
- підрахунок та індикацію IP-адрес, що доступні в заданий період.

Режими задавання відстежування внутрішньої обчислювальної мережі забезпечують:

- задавання конкретного числа IP-адреси абонента задля дослідження;
- задавання діапазону значень IP-адрес споживачів задля дослідження;
- задавання набору значень IP-адрес споживачів задля дослідження;
- задавання терміну, із якого розпочнеться дослідження споживачів мережі;
- задавання інтервалу терміну, на протязі якого відбудеться дослідження споживачів мережі;
- задавання інтервалу терміну поміж повторними запитами до споживачів внутрішньої обчислювальної мережі.

Режими задавання запис відстежування внутрішньої обчислювальної мережі забезпечують:

- запис всієї інформації щодо сканованих споживачів;
- запис інформації лише щодо споживачів, що змінювали свій стан.

Після запуску програмованого додатку задля діагностування сегментів обчислювальної мережі на екрані із'являється основна форма додатку, що містить візуальний вигляд (рис. 1.21).

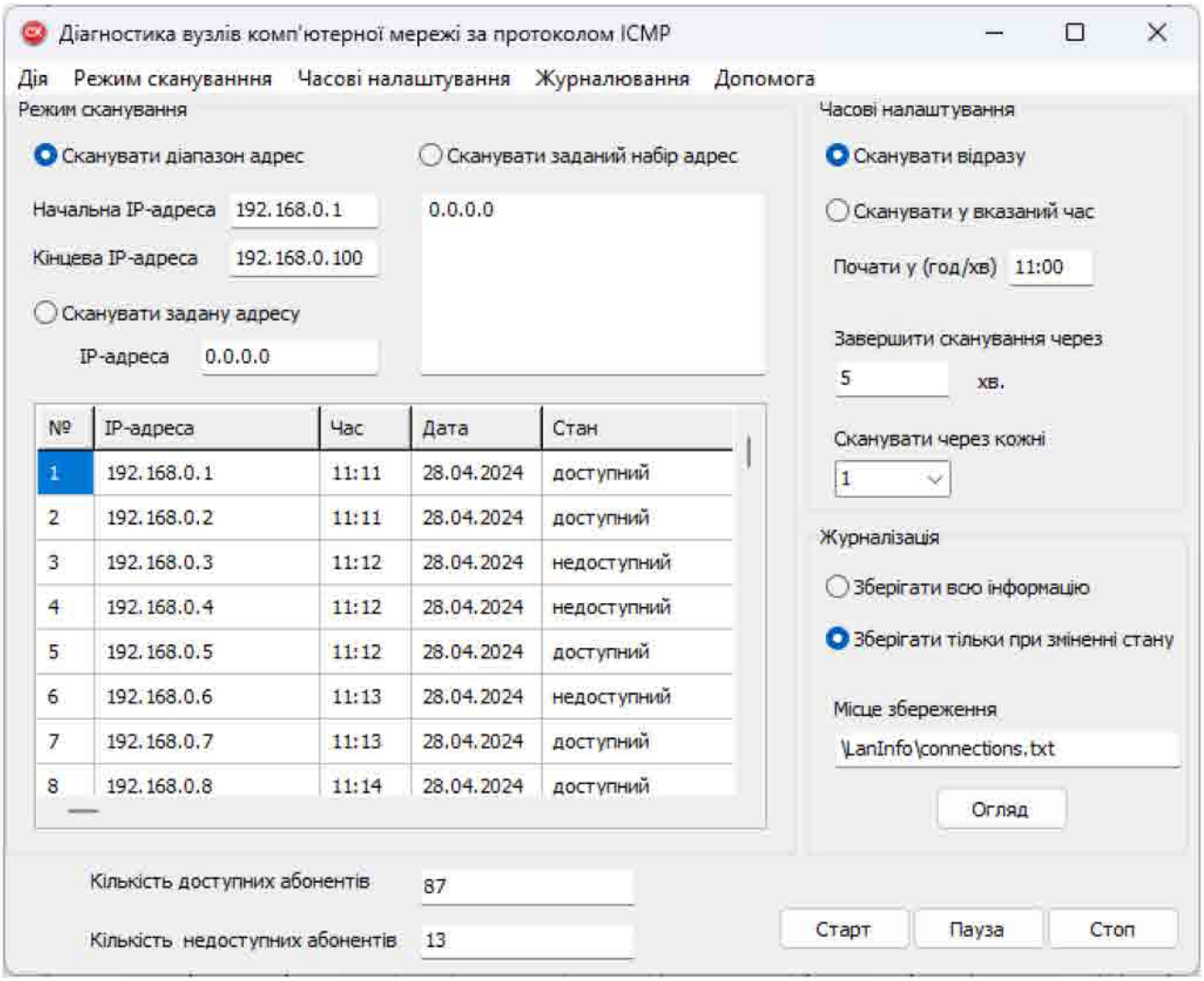



Рисунок 1.21. Головне вікно додатку діагностування сегментів мережі

В додатку передбачене головне набір, що представлене на рис. 1.22, та яке містить такі розділи: "Дія", "Режим дослідження", "Часові настройки", "Запис", "Допомога".

Розділ "Дія" складається із пунктів "Сканувати", "Пауза", "Стоп", "Вихід" (рис. 1.23).

При обиранні опції “Сканувати” розпочнеться хід дослідження заданих споживачів внутрішньої обчислювальної мережі.

При першому обиранні опції “Пауза” хід дослідження зупиниться до того терміну, коли користувач не вибере цей пункт знову.

Вибір опції “Стоп” зупинить хід дослідження споживачів із обраними параметрами самого процесу дослідження.

При обиранні опції “ Вихід ” програма відстежування споживачів внутрішньої обчислювальної мережі завершить свою роботу.

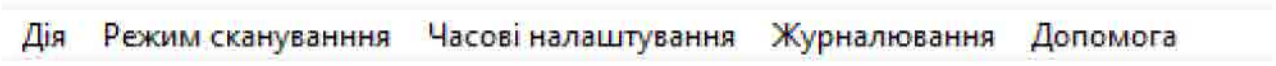



Рисунок 1.22. Набір додатку діагностування сегментів обчислювальної мережі

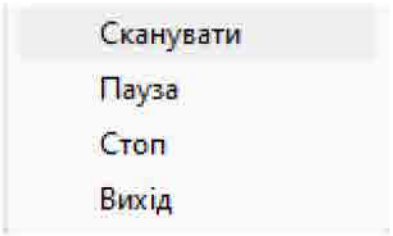



Рисунок 1.23. Розділ головного набір “Дія”

Розділ “Режим дослідження” складається із наступних пунктів: “Сканувати діапазон адрес”, “Сканувати задану адресу”, “Сканувати заданий набір адрес” (рис. 1.24).

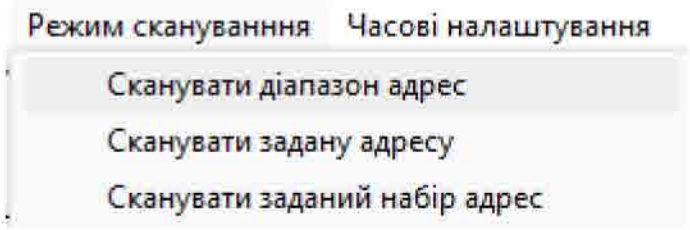



Рисунок 1.24. Розділ головного набір “Режим дослідження”

При обиранні опції “Сканувати діапазон адрес” користувачеві буде запропоновано обрати початкову і кінцеву IP-адресу споживачів внутрішньої обчислювальної мережі.

При виборі опції “Сканувати задану адресу” користувач повинен буде записати IP-адресу абонента, якого треба просканувати.

Коли користувач вибере пункт “Сканувати заданий набір адрес”, йому

треба буде записати всі необхідні IP-адреси споживачів, що потребують дослідження.

При записі IP-адреси треба її завдавати в вигляді послідовності чотирьох чисел в десятковій системі числення, котрі відділені поміж собою крапкою.

Розділ “Часові налаштування” складається із наступних пунктів: “Сканувати відразу”, “Сканувати в вказаний період”, “Період дослідження”, “Частота дослідження” (рис. 1.25).

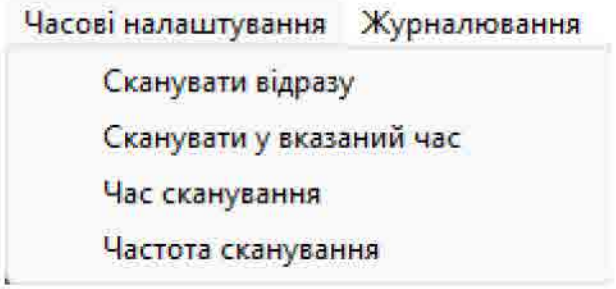



Рисунок 1.25. Розділ головного набір “Часові налаштування”

При виборі опції “Сканувати відразу” хід дослідження розпочнеться відразу після натискання кнопки “Старт” чи обирання відповідного опції набір.

При виборі опції “Сканувати в вказаний період” хід дослідження розпочнеться, коли настане заданий період.

Пункт “період дослідження” визначає, скільки терміну буде проходити хід дослідження споживачів внутрішньої обчислювальної мережі.

Після вибору опції “Частота дослідження” програма запропонує користувачеві обрати інтервал терміну, після якого треба знову провести операцію дослідження.

Розділ “Запис” допомагає зробити додаткові налаштування відносно процесу запис. Він складається із наступних пунктів: “Зберегти всю дані”, “Зберегти лише при зміненні стану”, але разом з цим “Місце збереження” (рис. 1.26).

При обиранні опції “Зберегти всю дані” зберігатися буде вся інформація відносно усіх споживачів, що спроможні знаходитися у діапазоні обраних IP-адрес дослідження.

При виборі опції “Зберегти лише при зміні стану” зберігатися буде лише

інформація щодо споживачів, що змінили свій стан в локальній обчислювальній мережі.

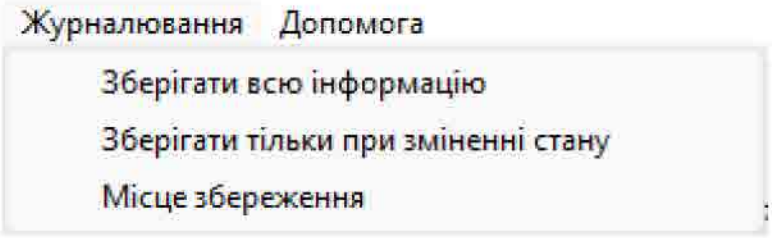



Рисунок 1.26. Розділ головного набір “Запис”

За поміччю опції “Місце збереження” користувач спроможне вказати місце і ім’я файлу запис.

Розділ “Допомога” відображає дані щодо програму (рис. 1.27).

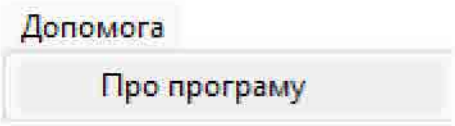



Рисунок 1.27. Розділ “Допомога”

При завданні режиму дослідження користувач буде обирати, сканувати діапазон адрес, чи сканувати конкретну IP-адресу, чи сканувати заданий набір IP-адрес. При цьому користувач буде вводити IP-адреси споживачів, котрі треба просканувати, в відповідні поля (рис. 1.28).

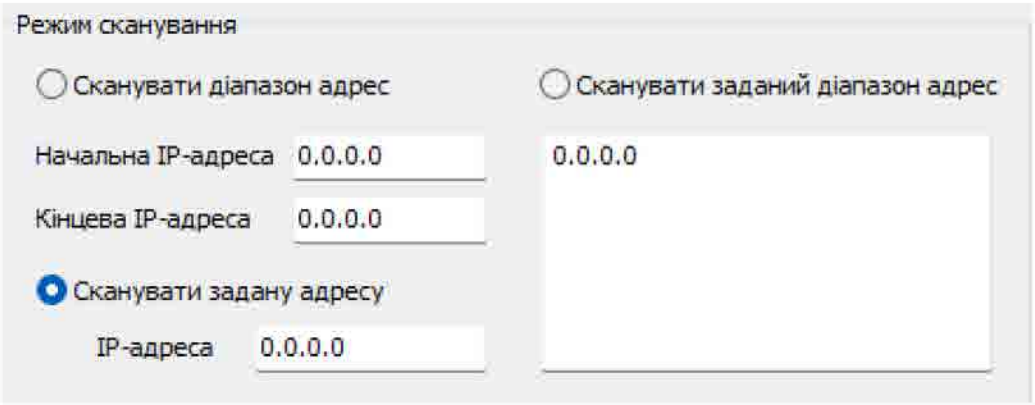



Рисунок 1.28. Вибір режиму дослідження

При занесені налаштувань терміну: початок дослідження, інтервал і тривалість, користувач буде користуватися частиною додатку, що представлена на рис. 1.29.

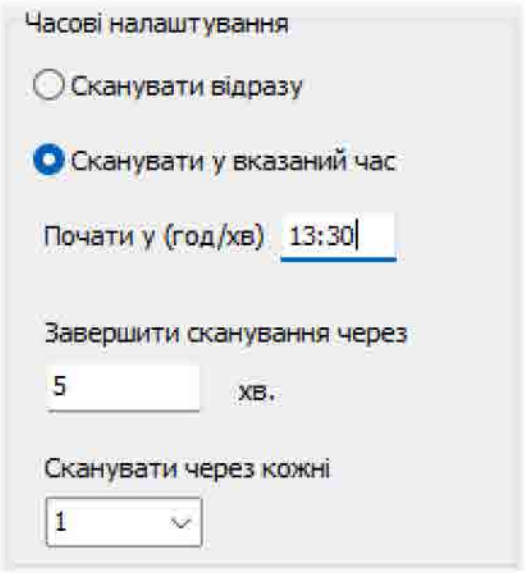



Рисунок 1.29. Налаштування терміну

Параметри задля запис процесів дослідження споживачів внутрішньої обчислювальної мережі користувач додатку буде вносити у поля, котрі відображені на рис. 1.30. Задля задавання місця знаходження файлу запис можливо натиснути кнопку “Огляд” (рис. 1.30).

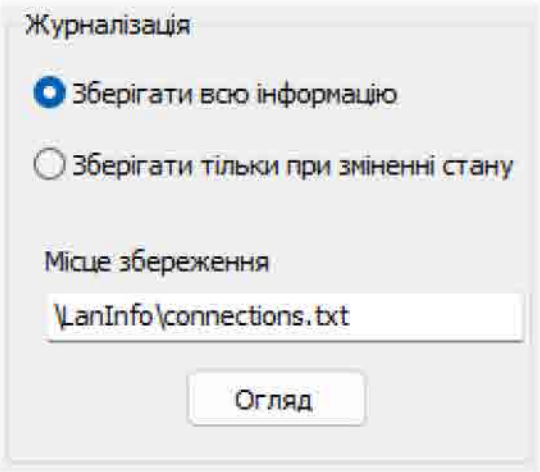



Рисунок 1.30. Запис

Керувати запуском, роботою і завершенням процесу дослідження споживачів мережі можливо за поміччю відповідних кнопок “Старт”, “Пауза”, “Стоп”, що знаходяться в нижньому правому куті основної форми додатку (рис. 1.31).

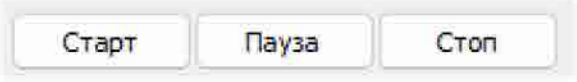



Рисунок 1.31. Кнопки керування процесом дослідження

Результати дослідження споживачів мережі записуються в файл запис і в відповідну таблицю, що відображена на рис. 1.32.

| № | IP-адреса | Час | Дата | Стан |
|---|---|---|---|---|
| 1 | | | | |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| 5 | | | | |
| 6 | | | | |
| 7 | | | | |
| 8 | | | | |

Рисунок 1.32. Таблиця відображення стану споживачів

Під період всього дослідження мережі підраховується кількість споживачів, що працюють в даний момент терміну і кількість споживачів, котрі недоступні відносно усіх сканованих споживачів внутрішньої обчислювальної мережі, що зазначені в режимі дослідження. Інформація щодо кількість працюючих і недоступних споживачів завжди відображається на головній формі додатку в вигляді, що представлений на рис. 1.33.

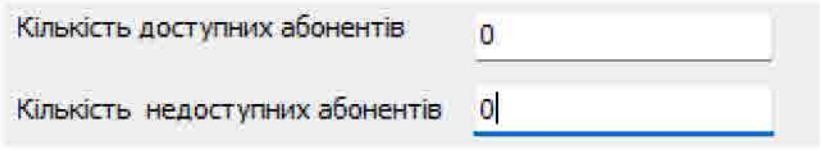



Рисунок 1.33. Інформація щодо споживачів

## 1.12 Підготовка до перевірка розробленого програмованого додатку

Перевірка працездатності комп'ютерних систем діагностування сегментів обчислювальної мережі виконана на тестових прикладах в частині:

- режимів вихідних установок додатку;
- виду демонстраційних елементів додатку;
- функціювання додатку в різних режимах;

- достовірності одержуваних результатів.

В вихідних установках моделі перевірені:

- правильність задавання IP-адрес;
- режимів функціювання додатку;
- значень вихідних інформації;
- режимів задавання терміну.

Демонстраційні елементи моделі перевірені на:

- правильність відображення IP-адрес споживачів ЛОМ;
- спроможність вірного відображення припустимих видів несправності внутрішньої обчислювальної мережі.

ІЗ точки зору функціювання моделі перевірена правильність обчислень, виконаних в розробленій програмі відстежування споживачів внутрішньої обчислювальної мережі:

- на конкретних значеннях IP-адрес;
- на конкретних значеннях наборів IP-адрес;
- на конкретних значеннях діапазонів IP-адрес;
- при різних часових налаштуваннях.

Достовірність одержуваних результатів перевіряється:

- порівнянням правильних значень знаходження споживачів в мережі із відповідними значеннями результатів, що обчислюються за помічю розробленого додатку відстежування внутрішньої обчислювальної мережі;
- візуальним спостереженням за роботою елементів керування та демонстраційних елементів додатку.

## 1.13 Випробування розроблених комп'ютерних систем

Під період випробування розробленого додатку було виконано дослідження внутрішньої обчислювальної мережі, до складу якої входили до 21 абонента, що були поділені на робочі групи. Застосунок був встановлений на сервері (21-му абоненті мережі). Конфігурація внутрішньої обчислювальної мережі, побудованої в САПР NetCracker, містить вигляд, показаний на рис. 1.34.

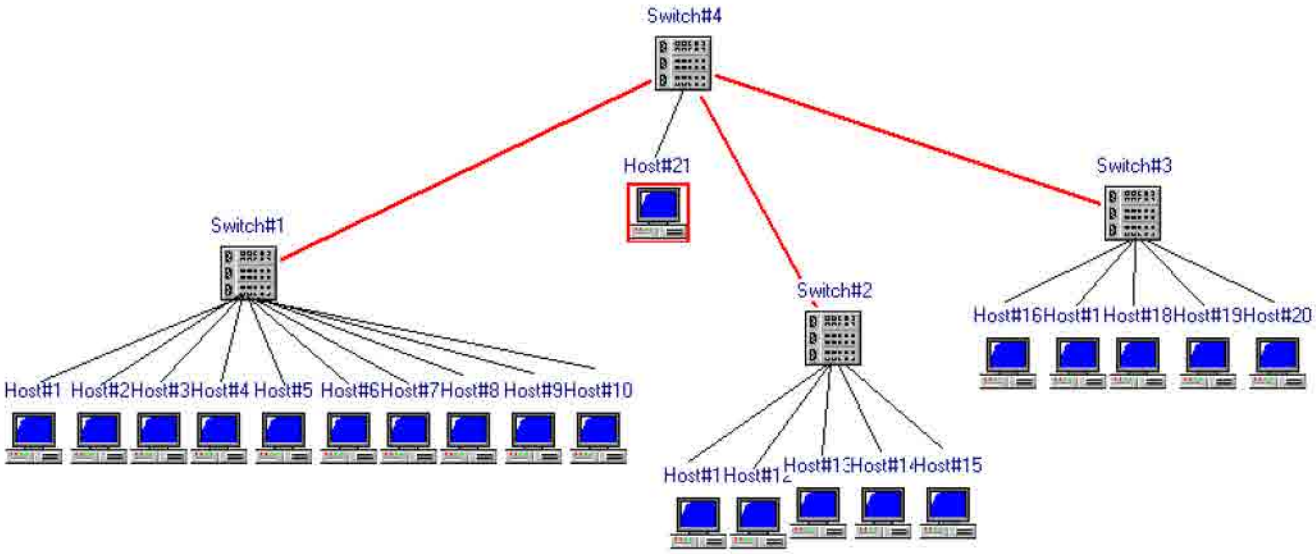



Рисунок 1.34. Побудова мережі задля випробування розробленого додатку

При випробуванні розроблених комп'ютерних систем діагностування сегментів обчислювальної мережі за протоколом I-C-M-P були отримані результати, наведені в табл. 1.1.

Таблиця 1.1. Результати діагностування сегментів обчислювальної мережі

| № | IP-адреса | Період | Дата | Стан |
|---|---|---|---|---|
| 1 | I92.I68.0.1 | 11:11 | 28.04.2024 | наявний |
| 2 | I92.I68.0.2 | 11:11 | 28.04.2024 | наявний |
| 3 | I92.I68.0.3 | 11:11 | 28.04.2024 | недоступний |
| 4 | I92.I68.0.4 | 11:12 | 28.04.2024 | недоступний |
| 5 | I92.I68.0.5 | 11:12 | 28.04.2024 | наявний |
| 6 | I92.I68.0.6 | 11:12 | 28.04.2024 | недоступний |
| 7 | I92.I68.0.7 | 11:12 | 28.04.2024 | наявний |
| 8 | I92.I68.0.8 | 11:12 | 28.04.2024 | наявний |
| 9 | I92.I68.0.9 | 11:13 | 28.04.2024 | наявний |
| IO | I92.I68.0.10 | 11:13 | 28.04.2024 | недоступний |
| 11 | I92.I68.0.11 | 11:13 | 28.04.2024 | наявний |
| 12 | I92.I68.0.12 | 11:13 | 28.04.2024 | наявний |
| 13 | I92.I68.0.13 | 11:13 | 28.04.2024 | наявний |
| 14 | I92.I68.0.14 | 11:13 | 28.04.2024 | наявний |
| 15 | I92.I68.0.15 | 11:13 | 28.04.2024 | недоступний |
| 16 | I92.I68.0.16 | 11:14 | 28.04.2024 | недоступний |

Всі дані, що були зібрані під період функціювання розробленого додатку відносно стану споживачів в локальній комп'ютерній обчислювальній мережі, збереглися в заданому текстовому файлі. Зберігання інформації в текстовому

файлі у форматі *.txt дає змогу адміністратору чи простому користувачеві відкривати того будь-яким редактором задля подальшого визначення. Так, зокрема, отримані дані можливо оброблювати іншими програмами, що дає змогу поєднати ці додатку в єдиний комплекс. Після функціювання додатку інформація в файлі запис містить вигляд, представлений на рис. 1.35.

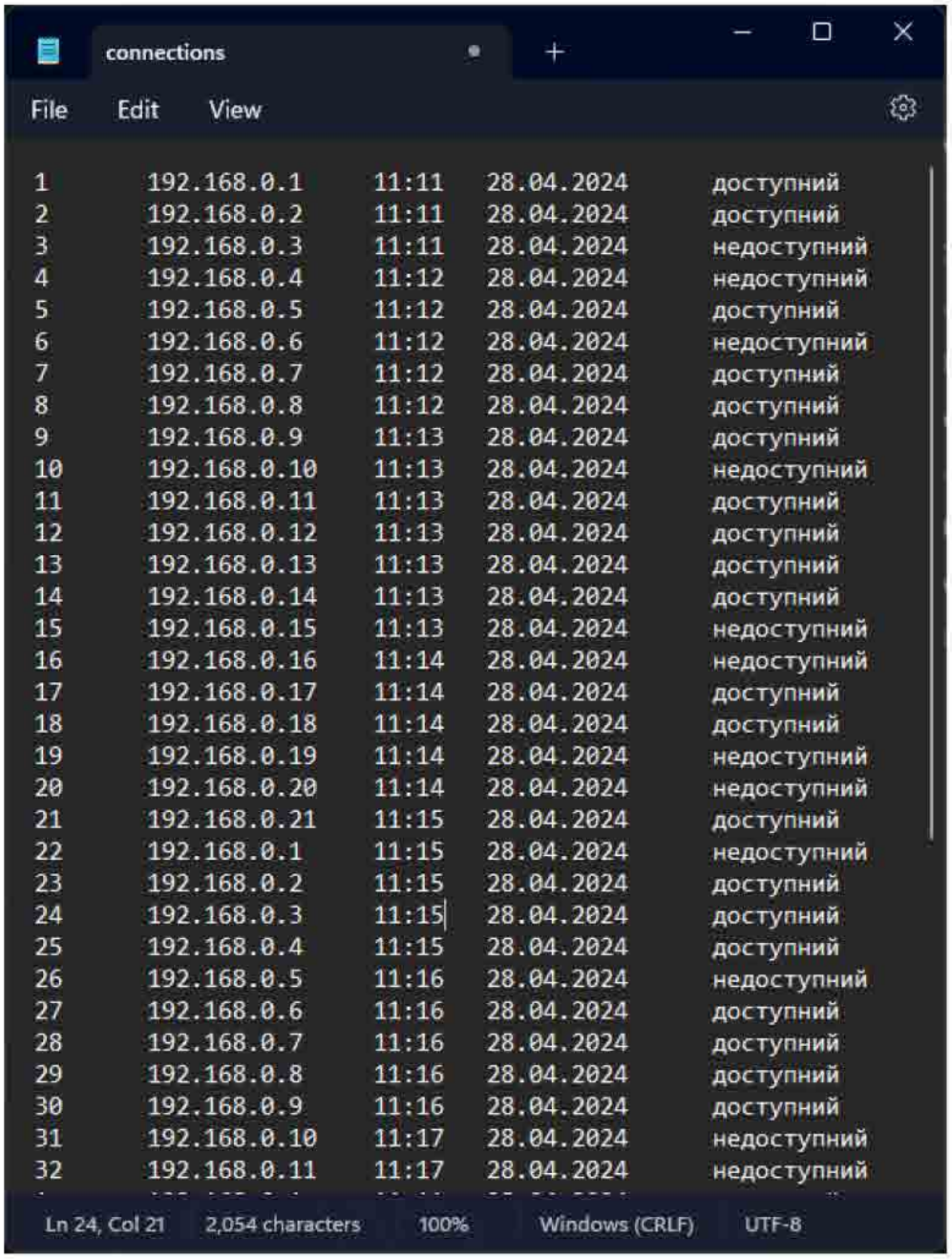



Рисунок 1.35. Файл запис стану сегментів обчислювальної мережі

За результатами, котрі були отримані за поміччю створеного додатку, були побудовані графіки працездатності споживачів в локальній обчислювальній мережі. Графіки відображають моменти терміну, коли абоненти були доступні і

коли до тих не було доступа (рис. 1.36).

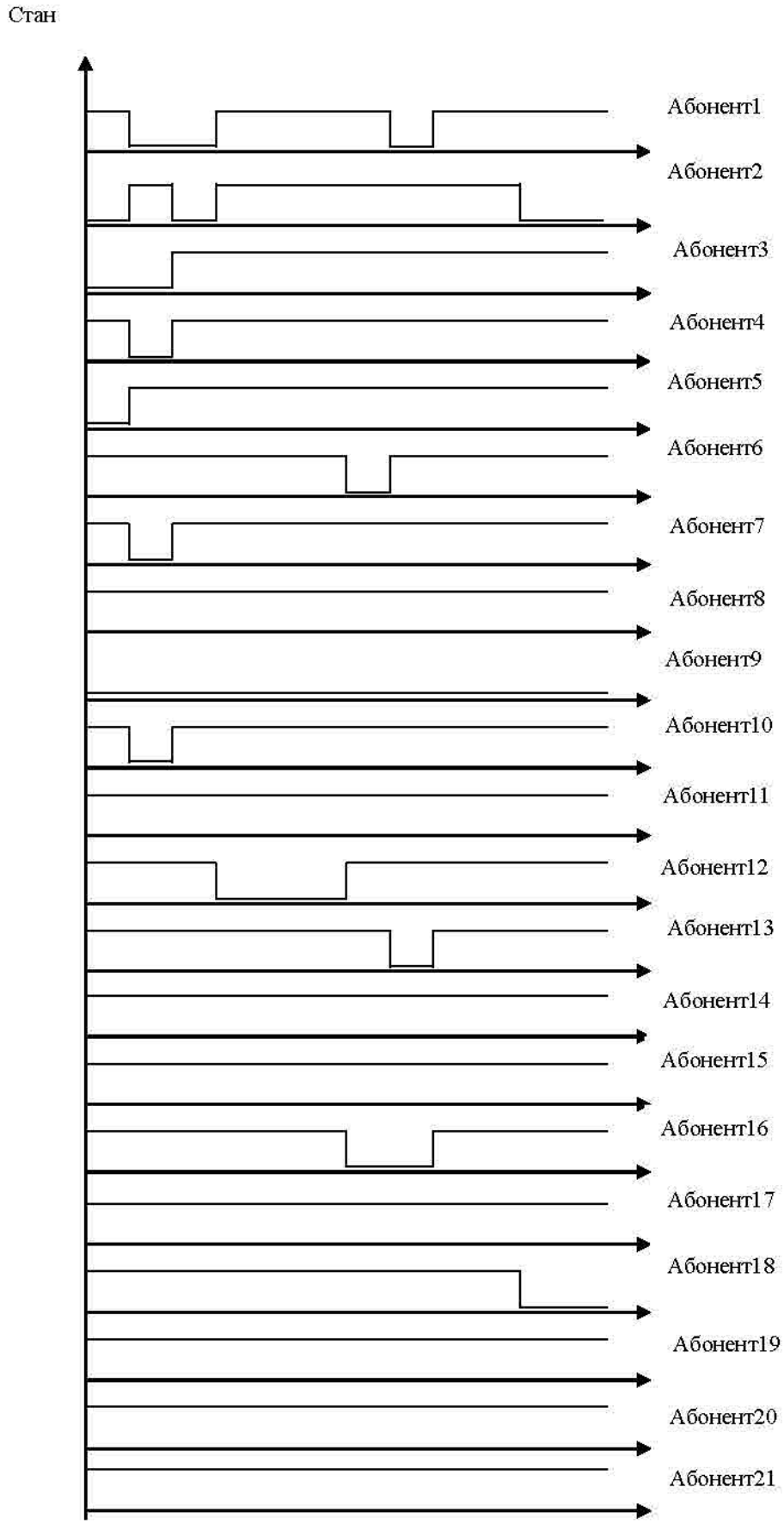



Рисунок 1.36. Графіки стану споживачів в мережі

Таким чином, за результатами визначення отриманих вище інформації видно, що частина споживачів не працює та при зверненні до тих спроможні виникати помилки при передачі інформації. Тім, таких помилок треба уникати. Інші абоненти працюють в нормальному режимі, тім передача інформації до тих можлива.

Отримані дані записуються в файл запис розробленого додатку. Є дає змогу адміністраторові і простому користувачеві дізнатися щодо стан доступа до конкретного абонента в будь-котрий проміжок терміну.

За поміччю розроблених комп'ютерних систем діагностування сегментів обчислювальної мережі за протоколом I-C-M-P легко визначити поточний стан конкретного абонента мережі. Застосунок відображує ці дані в спеціальній таблиці, що знаходиться в головному вікні, отже задля отримання такої інформації не потрібно звертатися до файлу запис.

# 2 ЕКОНОМІЧНИЙ РОЗДІЛ

## 2.1 Резюме

У даному дипломному проекті виконано розробку комп'ютерних систем діагностування сегментів обчислювальної мережі за протоколом I-C-M-P.

Ефективність кожного програмованого продукту визначається того якістю і ефективністю процесу розроблювання. Якість програм визначається наступними складовими: із точки зору споживача; із позиції застосування ресурсів; виконання вимог до програмованого середовища.

Оцінка якості програмованого продукту із точки зору споживача визначається необхідним на стадії функціювання розміром оперативної пам'яті ЕОТ, витратами машинного терміну, пропускною спроможністю каналів передачі інформації. Оцінка якості програмованого продукту включає визначення трудомісткості та вартості того створення.

Проведемо розрахунки визначення трудомісткості розроблювання даного програмованого продукту.

## 2.2 Розрахунок ціни програмованого продукту нормативним методом

### 2.2.1 Визначення трудомісткості розроблювання програмованого середовища

Тривалість розроблювання програмованого продукту залежить з того обсягу, трудомісткості розроблювання, кваліфікації виконавців, але разом з цим планових термінів, визначених умовами ринку.

Методом структурної аналогії у відповідних каталогах аналогів програмованого середовища визначається обсяг комп'ютерних систем, в тисячах умовних машинних команд додатку аналога.

В таблиці 2.1 представлені аналоги програмованого середовища, процедури яких, в більшому чи меншому ступені, виконує розроблений програмний продукт.

Таблиця 2.1

| Найменування програм | Обсяг процедури програм – $V_o$, усл. машинних командах. |
|---|---|
| 1. програм СУБД | 2500 – 9800 |
| 2. Комплексні моделі ведення БД | 950 – 7430 |
| 3. програм введення інформації | 1060 – 5750 |
| 4. програм оптимізації розрахунків | 1300 – 4200 |
| 5. програм автоматизації систем у каталогу | 680 – 7000 |
| 6. програм автоматизованих розрахунків | 1300 – 8600 |
| 7. програм загальної математики та програм імітаційного моделювання | 7800 – 8800 |
| 8. програм організації обчислювального процесу | 13000 – 10200 |

Задля нашого варіанта виділено сірим кольором.

Вибравши аналог програм, що містить Vo у умовних машинних командах, трудомісткості визначати на основі табл.2.2

Таблиця 2.2

| Обсяг програм, тис.умов. машинних команд | Норма терміну, люд/год |
|---|---|
| 1.00 | 229 |
| 2.00 | 244 |
| 3.00 | 262 |
| 4.00 | 283 |
| 5.00 | 306 |
| 6.00 | 330 |
| 7.00 | 357 |
| 8.00 | 385 |
| 9.00 | 414 |
| 10.00 | 445 |
| 12.00 | 510 |
| 14.00 | 580 |
| 16.00 | 654 |
| 18.00 | 731 |
| 20.00 | 812 |

На підставі отриманого числа, у довіднику, визначається укрупнена норма терміну на розробку аналога програмованого середовища (коректується поправочним коефіцієнтом враховуючої умови розроблювання програм, тобто у умовах комп'ютера, Кк=0,7÷0,8): Тар = 229 х 0,7 = 160,3 (люд/годин).

Трудомісткість програмованого продукту визначається у кожному етапу розроблювання окремо на підставі трудомісткості аналога із урахуванням складності розроблювання, ступеня новизни та ступеня застосування у розробці стандартних модулів на підставі формул:

$$T_{ТЗ} = T^a p \times L_1 \times K_H \quad (2.1)$$

$$T_{ТП} = T^a p \times L_2 \times K_H \quad (2.2)$$

$$T_{РП} = T^a p \times L_3 \times K_H \times K_T \quad (2.3)$$

Задля розрахунку необхідні наступні коефіцієнти:

Li – питома вага і-го етапу розроблювання (див. табл. 2.2);

Кн – поправочний коефіцієнт, що враховує ступінь новизни (див. табл. 2.3);

Кт – поправочний коефіцієнт, що враховує ступінь застосування у розробці типових програм (див. табл. 2.4).

Таблиця 2.3. Числа питомих коефіцієнтів трудомісткості стадії у загальній трудомісткості розроблювання програм

| Код стадії | Ступінь новизни | | |
|---|---|---|---|
| | АЛЕ | Б | У |
| ТЗ ($L_1$) | 0,15 | 0,12 | 0,12 |
| ТП ($L_2$) | 0,16 | 0,15 | 0,11 |
| РП ($L_3$) | 0,55 | 0,58 | 0,61 |

Задля нашого варіанта виділено сірим кольором.

Таблиця 2.4. Числа поправочного коефіцієнта, що враховує ступінь новизни

| Код ступеня новизни | Ступінь новизни | Числа $К_н$ |
|---|---|---|
| А | Принципово нове ПЗ | 1,75 – 1,2 |
| Б | ПЗ – розвиток визначеного параметричного ряду | 1,0 – 0,8 |
| В | ПЗ, що містить аналог | 0,7 |

Задля нашого варіанта виділено сірим кольором.

Тім що розробка моделі є ПЗ, що містить аналоги комп'ютерних продуктів, то код ступеня новизни задля мого ПЗ – У, але числа коефіцієнта $К_н$=0,7. У таблиці 2.3, знаючи код ступеня новизни, тепер можливо визначити числа питомих коефіцієнтів трудомісткості:

$L_1$=0,12;

$L_2$=0,11;

$L_3$=0,61;

Таблиця 2.5. Числа коефіцієнта ступеня застосування у розробці типових програм

| Ступінь охоплення реалізованих функцій розроблювального ПЗ типовими програмами, % | Числа $К_т$ |
|---|---|
| 60 та вище | 0,6 |
| 40-60 | 0,7 |
| 20-40 | 0,8 |
| До 20 | 0,9 |

Задля нашого варіанта виділено сірим кольором.

В розробленому програмному продукті використовується з 40 до 60 відсотків існуючих функцій, є значить, що $К_т$=0,7.

Тепер розраховуємо трудомісткість у кожному етапу окремо:

Трудомісткість технічного задавання

$$Т_{тз}=I*L_1*К_н=160,3 *0,12*0,8= 15,38 \text{ (люд/годин)} \qquad (2.4)$$

Трудомісткість розроблювання технічного проекту

$$T_{тп}=I*L_2*K_н=160,3\ *0,11*0,8 = 14,11 \text{ (люд/годин)} \quad (2.5)$$

Трудомісткість розроблювання робочого проекту

$$T_{рп}=I*L_3*K_н*K_т=160,3\ *0,61*0,8*0,8= 62,58 \text{ (люд/годин)} \quad (2.6)$$

Задля подальших розрахунків визначили кількість папера, витраченого на кожен етап:

- технічне задавання $N_{тз}$=3 (стр),

- розробка ТП $N_{тп}$=15(стр),

- розробка робочого проекту $N_{рп}$=20 (стр),

- пояснювальна записка відповідно $N_{пз}$=30 (стр)

Таблиця 2.6. Розрахунок трудомісткості програм

| Найменування етапів | Розрахунок, годин. | | |
|---|---|---|---|
| 1 | 2 | 3 | 4 |
| 1.ТЗ | $T_{Ртз}$=15,38 | $T_{кк}=0,7*N_{тз}$= 0,7*3=2,1 | $T_{нк}=0,15*N_{тз}$=0,15*3= 0,45 |
| 2.Розробка ТП | $T_{Ртп}$=14,11 | $T_{кк}=0,7*N_{тп}$=0,7*15=IO, 5 | $T_{нк}=0,15*N_{тп}$=0,15*15 =2,25 |
| 3.Розробка РП | $T_{Ррп}$= 62,58 | $T_{кк}=0,7*N_{рп}$=0,7*20=14, 0 | $T_{нк}=0,15*N_{рп}$=0,15*20 =3,0 |
| 4.Розробка ПЗ | $T_{пз}=1,5**N_{пз}$= 1,5*30 =45 | $T_{кк}=0,7*N_{тз}$=0,7*30=21, 0 | $T_{нк}=0,15*N_{пз}$=0,15*30 =4,5 |
| Усього, у т.ч.: | 194,87 | | |
| - на розробку | $\Sigma T_р$=137,07 | | |
| - контроль керівника | | $\Sigma T_{кк}$=47,6 | |
| - нормоконтроль | | | $\Sigma T_{нк}$=IO,2 |

# 3 РОЗДІЛ ОХОРОНИ ПРАЦІ І ТЕХНІКИ БЕЗПЕКИ

Роботодавець чи уповноважені ним органи зобов'язані дбати щодо умови праці працівників, полегшувати тих, оздоровляти навколишнє середовище, дбати щодо виконання правил безпеки та інструкцій у техніці безпеки.

Координує всю цю діяльність служба охорони праці, котра у залежності з чисельності працюючих спроможне функціонувати як самостійний структурний підрозділ (число працюючих 50 та більше), чи в вигляді групи спеціалістів чи одного спеціаліста, в тім числі за сумісництвом (число працюючих 20 та менше). Задачі службі охорони праці і її процедури викладені у Типовому положенні щодо службу охорони праці», яке затверджено наказом Комітетом Держнаглядохоронпраці (ДНАОП 0.00-4.21-93) .

Працівники разом з цим повинні відповідально ставитись до охорони праці, знати і виконувати вимоги, визначені нормативною документацією. У сучасних умовах кожному працівнику треба постійно підтримувати високий фізичний, психологічний і фаховий рівень, запобігати виникненню випадків травматизму і профзахворювань.

Безпечні умови праці на підприємстві досягаються за рахунок середовища безпеки виробничих процесів, котрі обґрунтовані та прийняті у технологічній частині дипломного проекту.

## 3.1 Аналіз небезпечних і шкідливих чинників, що впливають на працівника

Задля установлення можливого впливу на здоров'я користувачів ВДТ виробничих чинників містить числа ряд якісних характеристик робочого середовища. Є середовище в приміщеннях (офісах) у основному характеризується такими фізичними параметрами, як температура, вологість і електричний опір підлоги. Фізико-хімічні показники включають дані щодо вміст в повітрі іонів і різноманітних забруднювачів, але разом з цим деякі інші якісні характеристики середовища

## 3.2 Розробка заходів із охорони праці

### 3.2.1 Виробничі приміщення

Будівлі і приміщення, де розміщені робочі місця програмістів повинні відповідати вимогам СНіП 2.09.02-85 «Производственные здания» і ДСанПіН 3.3.2.007 – 98 «Державні санітарні правила та норми функціювання із ВДТ ЕОМ» Вони мають знаходитись не нижче другого ступеня вогнестійкості. Задля усіх приміщень повинно знаходитись визначено клас зони згідно із НПАОП 40.1-1.01-97. Відповідне позначення повинно знаходитись нанесено на вхідних дверях кожного приміщення.

Не дозволяється розташування приміщень із робочими місцями операторів РС в підвалах та цокольних поверхах. Площа приміщення із розрахунку на одне робоче місце містить знаходитись не менше 6,0 кв.м, але об'єм – не менше 20,0 куб.м.

Задля внутрішнього оздоблення приміщень із РС слід використовувати дифузно-відбивні матеріали із коефіцієнтом відбитті задля стелі 0,7 – 0,8, задля стін 0,5 – 0,6. Покриття підлоги повинне знаходитись матовим, поверхня рівною, не слизькою, із антистатичними властивостями.

Віконні прорізи приміщень задля функціювання із РС мають знаходитись обладнані регульованими пристроями (жалюзі, завіски, зовнішні козирки).

Забороняється задля оздоблення інтер'єру приміщень із РС застосовувати полімерні матеріали, що виділяють в повітря шкідливі хімічні речовини. Приміщення спроможні обладнуватись шафами задля зберігання документів, полицями, стелажами.

В приміщеннях слід щоденно робити вологе прибирання. Вони мають знаходитись оснащені аптечками першої медичної допомоги.

При приміщеннях із ВДТ мають знаходитись обладнані побутові приміщення задля відпочинку під період функціювання, кімната психологічного розвантаження, де слід передбачити встановлення систем задля приготування й роздачі тонізуючих напоїв, але разом з цим місця задля занять фізичною культурою ( СНиП 2.09.04 – 87).

### 3.2.2 Мікроклімат робочої зони працівників, вентиляція

Висока температура повітря негативно позначається на функціональному стані людини. Хоч генерація теплоти дисплеєм досягає критичного рівня лише в саму теплу пору року, треба створювати комфортні теплові умови постійно.

Оптимальні і допустимі мікрокліматичні параметри в приміщеннях повинні враховувати специфіку технологічного процесу при використанні комп'ютерів. Згідно із діючими в нашій країні нормативними документами ( ДСанПіН 3.3.2-007-98 в холодні періоди року температура повітря, швидкість того руху і відносна вологість повітря повинні відповідно складати: 22-24$^0$С; 0,1 м/с; 40-60%. Температура повітря спроможне коливатись в межах з 21 до 25$^0$С при збереженні інших параметрів мікроклімату.

У теплі періоди року температура повітря, того рухливість і відносна вологість повинні відповідно становити: 23-25$^0$С; 0,1-0,2 м/с; 40-60 %.

Оптимальним рівнем аероіонізації в зоні дихання споживача вважається вміст легких аерофонів обох знаків з 150 до 5000 в 1 см$^3$ повітря.

Нормалізуючий вплив на склад повітря робочої зони справляють примусова вентиляція, захисні екрани ( оснащені заземленням) і застосування іонізаторів.

### 3.2.3 Освітлення робочого місця, шум, вібрація

Освітлення в приміщеннях із ВДТ містить знаходитись змішаним – природним і штучним. Природне освітлення повинно здійснюватись в вигляді бічного освітлення і відповідати нормам ДБН У.2.5-28-2006 «Природне та штучне освітлення».

При природному освітленні слід передбачити наявність сонцезахисних систем, що знижують перепади яскравостей поміж природним світлом і свіченням екрана ВДТ. ІЗ цією метою можливо використовувати плівки із металізованим покриттям чи жалюзі із вертикальними лам елями, що регулюються.

Штучне освітлення в приміщеннях із ВДТ треба здійснювати в вигляді комбінованої моделі освітлення із використовуванням люмінесцентних джерел

світла в світильниках загального освітлення. На робочих місцях містить знаходитись забезпечена рівномірна освітленість за помічю переважно відбитого чи розсіяного світлорозподілу. Світлових відблисків із клавіатури, екрана і з інших частин ВДТ в напрямку очей споживача не повинно знаходитись.

Деякі ВДТ є потенційними джерелами цілого ряду звуків, що містять як коливання, котрі можливо почути, так та коливання ультразвукового діапазону. Цей шум справляє негативний вплив на стан споживача, особливо при тривалому впливі. Нормованим параметром шуму на робочих місцях є рівень 50 дБ. Основними заходами боротьби із шумом є усунення чи ослаблення причин шуму у самому того джерелі в процесі проектування, застосування систем звукопоглинання, раціональне планування виробничих приміщень.

### 3.2.4 Електробезпека

Причинами ураження працівника електрострумом спроможні знаходитись:

- Випадковий дотик до струмоведучих частин, в результаті ведення робіт поблизу чи на цих частинах;
- Випадковий дотик до струмоведучих частин, в результаті ведення робіт поблизу чи на цих частинах;
- Несправність захисних систем, якими потерпілий доторкався до струмоведучих частин;

Помилкове прийняття устаткування, що перебуває під Електробезпека.

Числа сили струму, що проходить через організм людини, залежить з напруги, під якою перебуває людина й з опору ділянки тіла, до якого прикладена ця напруга. Джерелом живлячої напруги є мережа змінного струму із напругою 229В, на яку поширюється ГОСТ 25861-83.

Основними причинами електротравматизму є:

- напругою, як відключеного;
- Несподіване виникнення напруги через ушкодження ізоляції там, де у нормальних умовах того знаходитись не повинно;
- Контакт струмопровідного устаткування із проводом, що перебуває під

напругою.

Задля попередження поразок електричним струмом треба чітко й в повному обсязі виконувати правила провадження робіт та правил технічної експлуатації. Треба виключити спроможність доступа оператора до частин устаткування, що працює під небезпечною напругою, до неізольованим частинам, призначеним задля функціювання при малій напрузі й не підключеним до захисного заземлення, але разом з цим підводити електроживлення до ПЕОМ з розетки за поміччю спеціальної вилки із заземлюючим контактом.

### 3.2.5 Організація робочого місця споживача РС

Устаткування та організація робочого місця із ВДТ мають забезпечувати відповідність конструкцій усіх елементів робочого місця і тих взаємного розташування, ергономічним вимогам, із урахуванням характеру та особливостей трудової діяльності ( ДСанПіН 3.3.2.-007-98).

Конструкція робочого місця й взаємне розташування усіх того елементів ( сидіння, органи керування, засобу відображення інформації) відповідають антропометричним, фізіологічним та психологічним вимогам, але разом з цим характеру функціювання. Конструкція робочих меблів дає спроможність забезпечувати спроможність індивідуального регулювання тих відповідно до потреб працівника задля підтримки зручної пози. Робочий стіл повинен знаходитись пофарбований матовою фарбою. Дисплей розташований так, що того верхній край перебуває на рівні очей, на відстані близько 70 см, що укладається у припустимі рамки з 60 до 90 см. Частота мерехтіння екрана дорівнює 100 Гц, що відповідає умові більше 70 Гц.

Задля зниження нервово-емоційного напруження, стомлювання, поліпшення мозкового кровообігу, подолання несприятливих наслідків гіподинамії, запобігання втомі доцільно впроваджувати виконання комплексу вправ, котрі наведені в Державних санітарних правилах та нормах функціювання із візуальними терміналами електронно-обчислювальних машин ДСанПіН 3.3.2.007-98

# ВИСНОВКИ

Налагодження внутрішньої обчислювальної мережі проводиться безпосередньо в процесі її експлуатації, внаслідок чого неминучим є виникнення неполадок: починаючи з виявлення потенційно нестійких ділянок мережі та закінчуючи логічними помилками конфігурації, спливаючими у ході функціювання.

В дипломній роботі виконано розробку комп'ютерних систем діагностування сегментів обчислювальної мережі і застосовано методику пошуку нестабільно працюючих сегментів в комп'ютерній мережі за поміччю правила керуючих чи контрольних повідомлень – ICMP, котрий є частиною правила IP. Методика передбачає одержувати оптимальні за критерієм продуктивність/вартість структури комутацій, як на стадії початкового проектування, так та на стадії модернізації.

Складене програмне середовища допомагає проводити моніторинг обчислювальної мережі на появу у ній помилок. Розроблений застосунок проводить дослідження доступа до усіх робочих станцій і серверів мережі і записує дані цих результатів в спеціальний файл. Є дозволить системному адміністратору не витрачати зайвий період на виявлення причини несправності мережі, але відразу приступити до розв'язання задачі. Отримані дані можливо оброблювати іншими програмами, що дає змогу поєднати ці додатку в єдиний комплекс, причому, інші додатку задля такого комплексу спроможні знаходитись написані сторонніми програмістами.

В роботі узагальнені та конкретизовані алгоритми відстежування стану обчислювальної мережі. Розроблений застосунок містить зручний задля споживача візуальний вигляд, що передбачає того застосовувати користувачами, котрі не мають глибоких знань в питаннях, пов'язаних із принципом функціювання правил і елементів обчислювальної мережі. В майбутньому передбачається робота над подальшим удосконаленням можливостей додатку задля середовища обліку різних ситуацій та обставин функціювання мережі (ЛОМ).

# ПЕРЕЛІК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ


1. Микитишин А.Г. Телекомунікаційні системи та мережі: навч. посіб. Тернопіль: ТНТУ ім. Івана Пулюя, 2017. – 384 с.
2. Кулаков Ю.О. «Комп'ютерні мережі», КПІ ім. Ігоря Сікорського, 2022.
3. Тарнавський Ю.А., Кузьменко І.М. «Організація комп'ютерних мереж», КПІ ім. Ігоря Сікорського, 2018.
4. Стрихалюк Б. М. Теорія побудови та протоколи інфокомунікаційних мереж: Конспект лекцій. – Львів: Львівська політехніка, 2017. – 121 с.
5. Хомуляк М.О. Адміністрування комп'ютерних систем і мереж/ Навчальний посібник. 2023 – 154 с.
6. Мосіюк О.О. Операційні системи та системне програмування: навчально-методичний посібник. Житомир: Вид-во ЖДУ ім. І. Франка, 2022. – 76 с.
7. Горський М.П. Прикладне програмування: від теорії до практики: навч. посіб.– Чернівці: Чернівец. нац. ун-т ім. Ю. Федьковича, 2021. – 120 с.
8. Ткачук В.М. Алгоритми і структура даних: Навч. посіб. / Івано-Франківськ: вид. Прикарпатського національного ун-ту, 2016. – 286 с.
9. Коноваленко І. В., Федорів П. С. Системне програмування у Windows з прикладами на Delphi. Тернопіль: ТНТУ ім. І. Пулюя, 2012. – 320 с.
10. Безменов М. І. Основи програмування у середовищі Delphi: навч. посіб. – Харків: НТУ «ХПІ», 2010. – 608 с.
11. Галісєєв Г. Системне програмування. – К.: Вид-во Університет "Україна", 2019. – 113 с.
12. Артамонов Є.Б., Системне програмування: Лабораторний практикум / Є.Б. Артамонов, Г.М. Кременецький – К.: НАУ, 2017. – 80 с.
13. Embarcadero Delphi: веб-сайт.
    URL: https://uk.wikipedia.org/wiki/Embarcadero_Delphi.
14. Embarcadero RAD Studio: веб-сайт.
    URL: https://uk.wikipedia.org/wiki/Embarcadero_RAD_Studio.
15. Net Cracker 4.1. User Manual: веб-сайт.
    http://soft-landia.ru/netcracker.html. – NetCracker Professional 4.1.

# ДОДАТОК А

## Лістинг головного модулю застосунку для діагностики вузлів комп'ютерної мережі за протоколом ICMP (Embarcadero RAD Studio)

```
unit icmp;
interface
uses windows, unit1;
Const
// IP_STATUS codes returned from IP APIs
IP_STATUS_BASE = 11000;
IP_SUCCESS = 0;
IP_BUF_TOO_SMALL = (IP_STATUS_BASE + 1);
IP_DEST_NET_UNREACHABLE = (IP_STATUS_BASE + 2);
IP_DEST_HOST_UNREACHABLE = (IP_STATUS_BASE + 3);
IP_DEST_PROT_UNREACHABLE = (IP_STATUS_BASE + 4);
IP_DEST_PORT_UNREACHABLE = (IP_STATUS_BASE + 5);
IP_NO_RESOURCES = (IP_STATUS_BASE + 6);
IP_BAD_OPTION = (IP_STATUS_BASE + 7);
IP_HW_ERROR = (IP_STATUS_BASE + 8);
IP_PACKET_TOO_BIG = (IP_STATUS_BASE + 9);
IP_REQ_TIMED_OUT = (IP_STATUS_BASE + 10);
IP_BAD_REQ = (IP_STATUS_BASE + 11);
IP_BAD_ROUTE = (IP_STATUS_BASE + 12);
IP_TTL_EXPIRED_TRANSIT = (IP_STATUS_BASE + 13);
IP_TTL_EXPIRED_REASSEM = (IP_STATUS_BASE + 14);
IP_PARAM_PROBLEM = (IP_STATUS_BASE + 15);
IP_SOURCE_QUENCH = (IP_STATUS_BASE + 16);
IP_OPTION_TOO_BIG = (IP_STATUS_BASE + 17);
IP_BAD_DESTINATION = (IP_STATUS_BASE + 18);
// The next group are status codes passed up on status indications to
// transport layer protocols.
IP_ADDR_DELETED = (IP_STATUS_BASE + 19);
IP_SPEC_MTU_CHANGE = (IP_STATUS_BASE + 20);
IP_MTU_CHANGE = (IP_STATUS_BASE + 21);
IP_UNLOAD = (IP_STATUS_BASE + 22);
IP_GENERAL_FAILURE = (IP_STATUS_BASE + 50);
MAX_IP_STATUS = IP_GENERAL_FAILURE;
IP_PENDING = (IP_STATUS_BASE + 255);
// Values used in the IP header Flags field.
IP_FLAG_DF = $2; // Don't fragment this packet.
// Supported IP Option Types.
// These types define the options which may be used in the OptionsData field
// of the ip_option_information structure. See RFC 791 for a complete
// description of each.
IP_OPT_EOL = 0; // End of list option
IP_OPT_NOP = 1; // No operation
IP_OPT_SECURITY = $82; // Security option
IP_OPT_LSRR = $83; // Loose source route
IP_OPT_SSRR = $89; // Strict source route
IP_OPT_RR = $7; // Record route
IP_OPT_TS = $44; // Timestamp
IP_OPT_SID = $88; // Stream ID (obsolete)
MAX_OPT_SIZE = 40; // Maximum length of IP options in bytes
Type
```

```
TIPAddr=integer; // An IP address.
TIPMask=integer; // An IP subnet mask.
TIP_STATUS=Integer; // Status code returned from IP APIs.
POption_Information=^TOption_Information;
TOption_Information=record
Ttl:byte; // Time To Live
Tos:byte; // Type Of Service
Flags:byte; // IP header flags
OptionsSize:byte; // Size in bytes of options data
OptionsData:pointer; // Pointer to options data
end;
Picmp_echo_reply=^Ticmp_echo_reply;
Ticmp_echo_reply=record
Address:TipAddr; // Replying address
Status:integer; // Reply IP_STATUS
RoundTripTime:integer; // RTT in milliseconds
DataSize:word; // Reply data size in bytes
Reserved:word; // Reserved for system use
Data:pointer; // Pointer to the reply data
Options:Toption_Information; // Reply options
end;
TsmICMP_Echo_Reply=record
Address:TipAddr; // Replying address
Status:integer; // Reply IP_STATUS
RoundTripTime:integer; // RTT in milliseconds
DataSize:word; // Reply data size in bytes
Reserved:word; // Reserved for system use
DataPtr:pointer; // Pointer to the reply data
Options:Toption_Information; // Reply options
Data: array [0..255] of Char;
end;

function IcmpCreateFile:Thandle; StdCall;
function IcmpCloseHandle (H:Thandle):Bool; StdCall;
function IcmpSendEcho (IcmpHandle:Thandle;DestinationAddress:TipAddr;
RequestData:pointer;RequestSize:word;
RequestOptions:POption_Information;ReplyBuffer:pointer;
ReplySize:integer;Timeout:integer):Integer; stdcall;
Implementation
function IcmpCreateFile; external 'Icmp.Dll';
function IcmpCloseHandle; external 'Icmp.Dll';
Function IcmpSendEcho; external 'Icmp.Dll';
end.

unit pingModule;
interface
uses icmp, Windows;
const
INADDR_NONE: integer = -1;
procedure Ping (const Address, EchoString: PChar;var PingReply: TsmICMP_Echo_Reply;
const PingTimeout: Integer = 500);
function PingStatusToStr (StatusCode: integer): string;
function inet_addr (IPAddress: PChar): TipAddr; StdCall;
implementation
uses Dialogs, SysUtils;
```

```
procedure Ping (const Address, EchoString: PChar;
var PingReply: TsmICMP_Echo_Reply;
const PingTimeout: Integer = 500);
var
IPAddress: TipAddr;
ICMPPort: THandle;
begin
IPAddress: = inet_addr (Address);
if (IPAddress = INADDR_NONE) then
begin
raise Exception.Create ('Function call inet_addr failed. ' +
'The IP address is probably invalid.');
end;
ICMPPort: = IcmpCreateFile ();
if (ICMPPort = INVALID_HANDLE_VALUE) then
begin
raise Exception.Create ('Function call IcmpCreateFile failed.');
end;
IcmpSendEcho (ICMPPort, IPAddress,
EchoString, Length (EchoString), nil,
@PingReply, SizeOf (PingReply), PingTimeout);
IcmpCloseHandle (ICMPPort);
end;

function PingStatusToStr (StatusCode: integer): string;
begin
case (StatusCode) of
IP_SUCCESS: Result: = 'IP_SUCCESS';
IP_BUF_TOO_SMALL: Result: = 'IP_BUF_TOO_SMALL';
IP_DEST_NET_UNREACHABLE: Result: = 'IP_DEST_NET_UNREACHABLE';
IP_DEST_HOST_UNREACHABLE: Result: = 'IP_DEST_HOST_UNREACHABLE';
IP_DEST_PROT_UNREACHABLE: Result: = 'IP_DEST_PROT_UNREACHABLE';
IP_DEST_PORT_UNREACHABLE: Result: = 'IP_DEST_PORT_UNREACHABLE';
IP_NO_RESOURCES: Result: = 'IP_NO_RESOURCES';
IP_BAD_OPTION: Result: = 'IP_BAD_OPTION';
IP_HW_ERROR: Result: = 'IP_HW_ERROR';
IP_PACKET_TOO_BIG: Result: = 'IP_PACKET_TOO_BIG';
IP_REQ_TIMED_OUT: Result: = 'IP_REQ_TIMED_OUT';
IP_BAD_REQ: Result: = 'IP_BAD_REQ';
IP_BAD_ROUTE: Result: = 'IP_BAD_ROUTE';
IP_TTL_EXPIRED_TRANSIT: Result: = 'IP_TTL_EXPIRED_TRANSIT';
IP_TTL_EXPIRED_REASSEM: Result: = 'IP_TTL_EXPIRED_REASSEM';
IP_PARAM_PROBLEM: Result: = 'IP_PARAM_PROBLEM';
IP_SOURCE_QUENCH: Result: = 'IP_SOURCE_QUENCH';
IP_OPTION_TOO_BIG: Result: = 'IP_OPTION_TOO_BIG';
IP_BAD_DESTINATION: Result: = 'IP_BAD_DESTINATION';
IP_ADDR_DELETED: Result: = 'IP_ADDR_DELETED';
IP_SPEC_MTU_CHANGE: Result: = 'IP_SPEC_MTU_CHANGE';
IP_MTU_CHANGE: Result: = 'IP_MTU_CHANGE';
IP_UNLOAD: Result: = 'IP_UNLOAD';
IP_GENERAL_FAILURE: Result: = 'IP_GENERAL_FAILURE';
else
Result: = '';
end;
end;
```

```
function inet_addr; external 'WSock32.Dll';
end.

    N7: TMenuItem;
    N8: TMenuItem;
    N9: TMenuItem;
    N10: TMenuItem;
    N11: TMenuItem;
    N12: TMenuItem;
    N13: TMenuItem;
    N14: TMenuItem;
    N15: TMenuItem;
    N16: TMenuItem;
    N17: TMenuItem;
    N18: TMenuItem;
    N19: TMenuItem;
    N20: TMenuItem;
    Label13: TLabel;
    Label14: TLabel;
    Edit7: TEdit;
    Edit8: TEdit;
procedure Button1Click (Sender: TObject);
procedure IdIcmpClient1Reply (ASender: TComponent;
const AReplyStatus: TReplyStatus);
    procedure Image1Click(Sender: TObject);
private
{Private declarations}
public
{Public declarations}
end;

var
Form1: TForm1;

implementation
uses pingModule,icmp;
{$R *.dfm}

procedure TForm1.Button1Click (Sender: TObject);
var Reply:TsmICMP_Echo_Reply;
begin
Drawgrid1.ColWidths[0]:=20;
Drawgrid1.ColWidths[1]:=100;
Drawgrid1.ColWidths[2]:=64;
Drawgrid1.ColWidths[3]:=64;
Drawgrid1.ColWidths[4]:=84;
//Drawgrid1.;

Self.IdIcmpClient1.Host:='localhost';
Self.IdIcmpClient1.TTL:=192;
Self.IdIcmpClient1.Ping;
Ping ('192.168.0.2',nil,Reply,10000);
ListBox1.Items.Add ('RawReply:'+IntToStr (Reply.RoundTripTime));
end;
```

```
procedure TForm1.IdIcmpClient1Reply (ASender: TComponent;
const AReplyStatus: TReplyStatus);
begin
ListBox1.Items.Add ('Reply:'+IntToStr (AReplyStatus.MsRoundTripTime));
end;

procedure TForm1.Image1Click(Sender: TObject);
var i:integer;
begin
 for I:= 1 to 255 do
begin
Ping (i,nil,Reply,10000);
ListBox1.Items('Reply:'+IntToStr (AReplyStatus.MsRoundTripTime));

end;
end;
end.

        type
          ip_option_information = packed record
        // Інформація заголовку IP (Наповнення
   // цієї структури і формат полів описано у RFC791.
            Ttl : byte;
        // Час життя (використовується traceroute-ом)
            Tos : byte;
          // Тип обслуговування, звичайно 0
            Flags : byte;
          // Прапори заголовку IP, звичайно 0
            OptionsSize : byte;
        // Розмір даних у заголовку, звичайно 0, максимум 40
            OptionsData : Pointer;
        // Вказівник на дані
          end;

          icmp_echo_reply = packed record
            Address : u_long;
         // Адреса відповідаючого
            Status : u_long;
        // IP_STATUS (див. нижче)
            RTTime : u_long;
        // Час між ехо-запитом і ехо-відповіддю
          // у мілісекундах
            DataSize : u_short;
         // Розмір повернених даних
            Reserved : u_short;
         // Зарезервовано
            Data : Pointer;
        // Вказівник на повернені дані
            Options : ip_option_information;
        // Інформація із заголовку IP
          end;

          PIPINFO = ^ip_option_information;
          PVOID = Pointer;
```

```
    function IcmpCreateFile() : THandle; stdcall; external 'ICMP.DLL' name
'IcmpCreateFile';

    function IcmpCloseHandle(IcmpHandle : THandle) : BOOL; stdcall; external
'ICMP.DLL'  name 'IcmpCloseHandle';

 function IcmpSendEcho(
   IcmpHandle : THandle;    // handle, повернений IcmpCreateFile()
DestAddress : u_long;    // Адреса отримувача (у мережевому порядку)
 RequestData : PVOID;     // Вказівник на надіслані дані
                  RequestSize : Word;      // Розмір надсилаємих даних
                  RequestOptns : PIPINFO;
// Вказівник на надсилаєму структуру
  // ip_option_information (может быть nil)
                  ReplyBuffer : PVOID;
// Вказівник на буфер, що містить відповіді
                  ReplySize : DWORD;
 // Розмір буфера відповідей
                  Timeout : DWORD
// Час очікування відповіді у милісекундах
                ) : DWORD; stdcall; external 'ICMP.DLL' name 'IcmpSendEcho';


{Функція IcmpSendEcho() надсилає ICMP ехо-запит за заданою IP-адресою і
поміщує всі відповіді, отримані за час заданого таймаута, у буфер віповідей
(ReplyBuffer).} WSAStartup() з Winsock.
 IcmpSendEcho()

(WSASYSNOTREADY).

{Можливі значення поля Status структури icmp_echo_reply} (IP_STATUS):

IP_STATUS_BASE 11000
IP_SUCCESS 0
IP_BUF_TOO_SMALL 11001
IP_DEST_NET_UNREACHABLE 11002
IP_DEST_HOST_UNREACHABLE 11003
IP_DEST_PROT_UNREACHABLE 11004
IP_DEST_PORT_UNREACHABLE 11005
IP_NO_RESOURCES 11006
IP_BAD_OPTION 11007
IP_HW_ERROR 11008
IP_PACKET_TOO_BIG 11009
IP_REQ_TIMED_OUT 11010
IP_BAD_REQ 11011
IP_BAD_ROUTE 11012
IP_TTL_EXPIRED_TRANSIT 11013
IP_TTL_EXPIRED_REASSEM 11014
IP_PARAM_PROBLEM 11015
IP_SOURCE_QUENCH 11016
IP_OPTION_TOO_BIG 11017
IP_BAD_DESTINATION 11018
IP_ADDR_DELETED 11019
IP_SPEC_MTU_CHANGE 11020
```

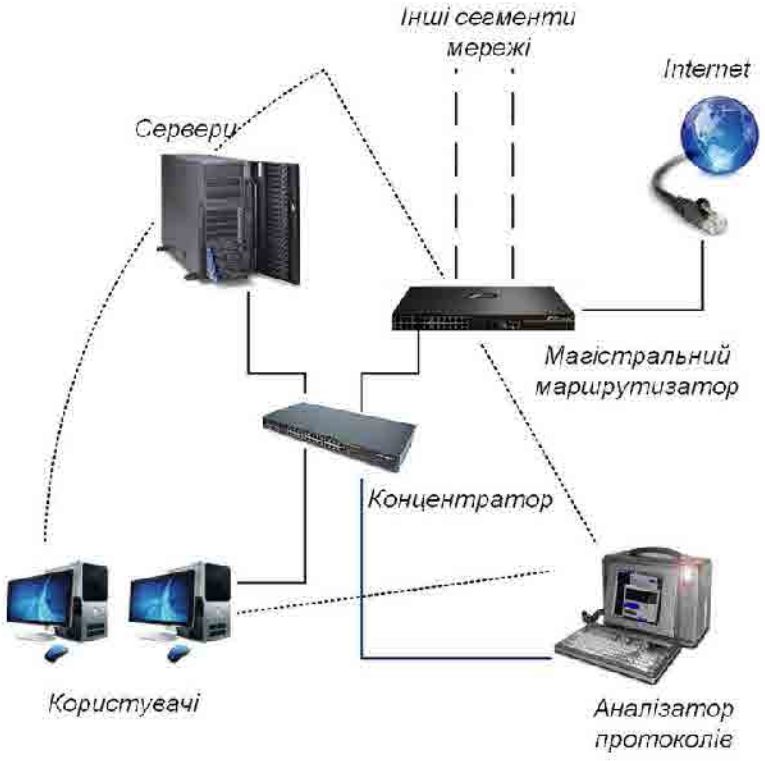
```
IP_MTU_CHANGE 11021
IP_UNLOAD 11022
IP_GENERAL_FAILURE 11050
MAX_IP_STATUS IP_GENERAL_FAILURE
IP_PENDING 11255

{Нижче наведений базовий фрагмент коду, необхідний для однократного
посилання-запиту: }

procedure TForm1.Button1Click(Sender: TObject);
var
   hIP : THandle;
   pingBuffer : array [0..31] of Char;
   pIpe : ^icmp_echo_reply;
   pHostEn : PHostEnt;
   wVersionRequested : WORD;
   lwsaData : WSAData;
   error : DWORD;
   destAddress : In_Addr;
begin

   // Створюємо handle
   hIP := IcmpCreateFile();

   GetMem( pIpe,
         sizeof(icmp_echo_reply) + sizeof(pingBuffer));
   pIpe.Data := @pingBuffer;
   pIpe.DataSize := sizeof(pingBuffer);

   wVersionRequested := MakeWord(1,1);
   error := WSAStartup(wVersionRequested,lwsaData);
   if (error <> 0) then
   begin
      Memo1.SetTextBuf('Error in call to '+
                  'WSAStartup().');
      Memo1.Lines.Add('Error code: '+IntToStr(error));
      Exit;
   end;

   pHostEn := gethostbyname('delphi.mastak.com');
   error := GetLastError();
   if (error <> 0) then
   begin
      Memo1.SetTextBuf('Error in call to'+
                  'gethostbyname().');
      Memo1.Lines.Add('Error code: '+IntToStr(error));
      Exit;
   end;

    destAddress := PInAddr(pHostEn^.h_addr_list^)^;

     // Посилаємо ping-пакет
   Memo1.Lines.Add('Pinging ' +
               pHostEn^.h_name+' ['+
               inet_ntoa(destAddress)+'] '+
```

```
                ' with '+
                IntToStr(sizeof(pingBuffer)) +
                ' bytes of data:');

        IcmpSendEcho(hIP,
                destAddress.S_addr,
                @pingBuffer,
                sizeof(pingBuffer),
                Nil,
                pIpe,
                sizeof(icmp_echo_reply) + sizeof(pingBuffer),
                5000);

        error := GetLastError();
        if (error <> 0) then
        begin
           Memo1.SetTextBuf('Error in call to '+
                    'IcmpSendEcho()');
           Memo1.Lines.Add('Error code: '+IntToStr(error));
           Exit;
        end;

         // Перевіряємо деякі з даних, що повернулися
        Memo1.Lines.Add('Reply from '+
                IntToStr(LoByte(LoWord(pIpe^.Address)))+'.'+
                IntToStr(HiByte(LoWord(pIpe^.Address)))+'.'+
                IntToStr(LoByte(HiWord(pIpe^.Address)))+'.'+
                IntToStr(HiByte(HiWord(pIpe^.Address))));
        Memo1.Lines.Add('Reply time: '+IntToStr(pIpe.RTTime)+' ms');

        IcmpCloseHandle(hIP);
        WSACleanup();
        FreeMem(pIpe);
      end;
//---------------------------------------------------------------------
```

# ДОДАТОК Б. Слайди мультимедійної презентації

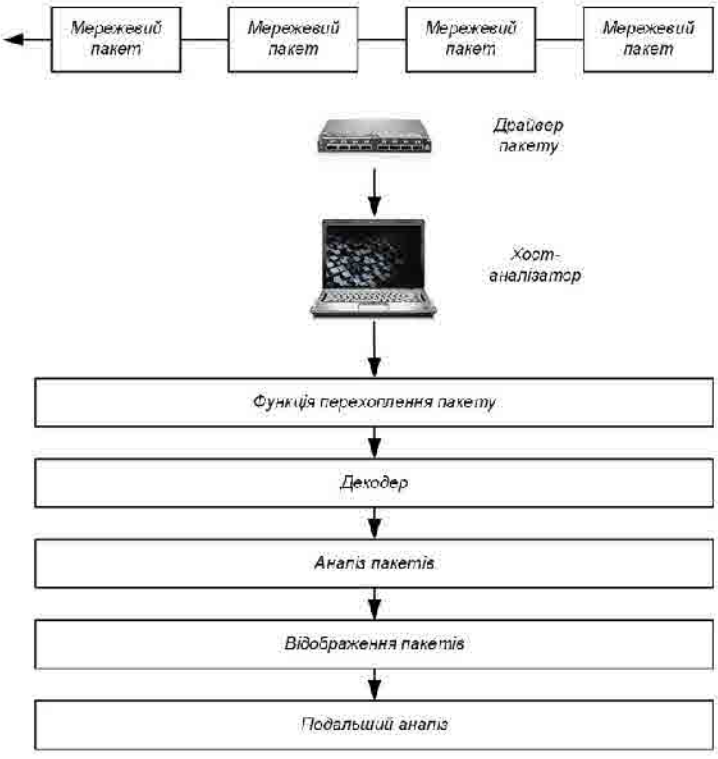
## Розробка програмних засобів діагностики вузлів комп'ютерної мережі за протоколом ICMP

Васильєв Вадим, гр. 4КС-57
ВСП «ОТФК ОНТУ»



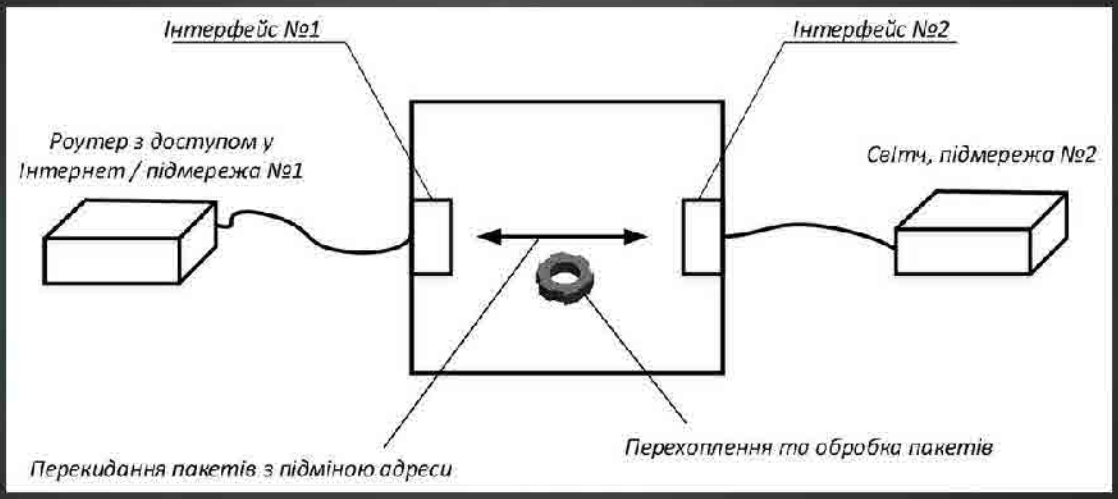

Традиційна локальна мережа з середовищем передачі, що розділяється, і аналізатором протоколів


# Комутована мережа з аналізатором протоколів



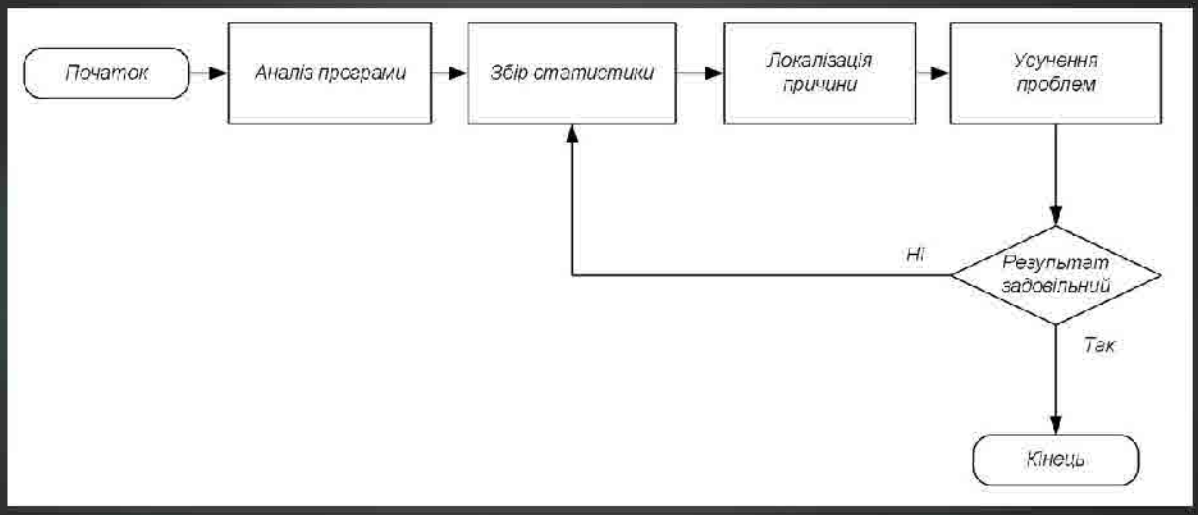

# Аналізатор протоколів виконує моніторинг мережевого трафіку

# Узагальнена схема структури мережі при роботі з розробленим ПЗ

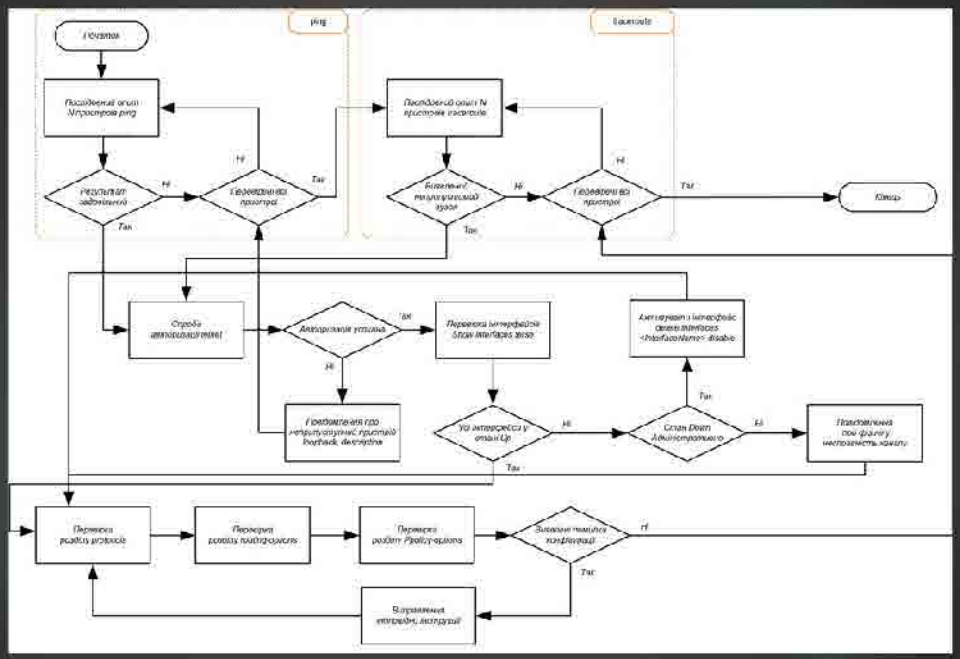



# Схема стенду для тестування розробленого ПЗ

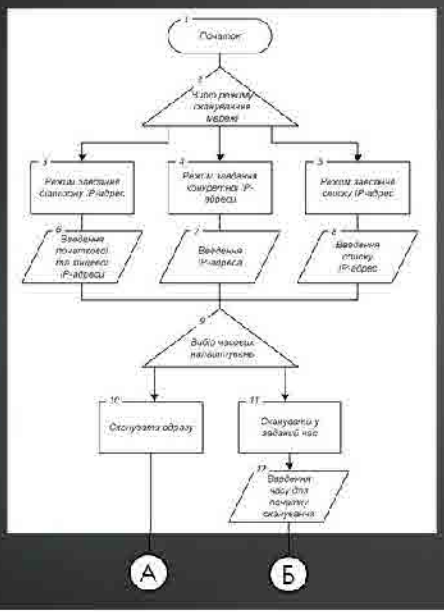
# Узагальнений алгоритм пошуку несправностей у мережі

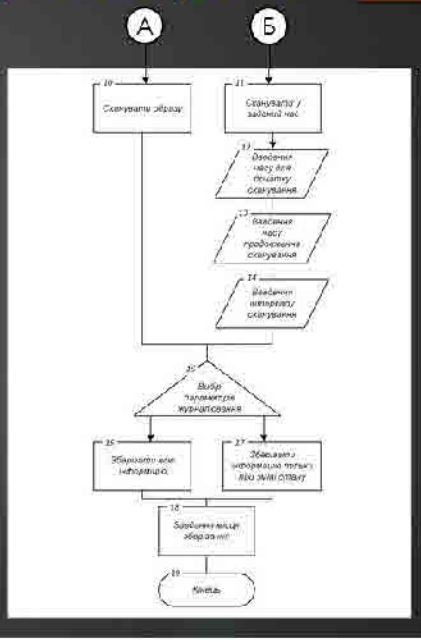


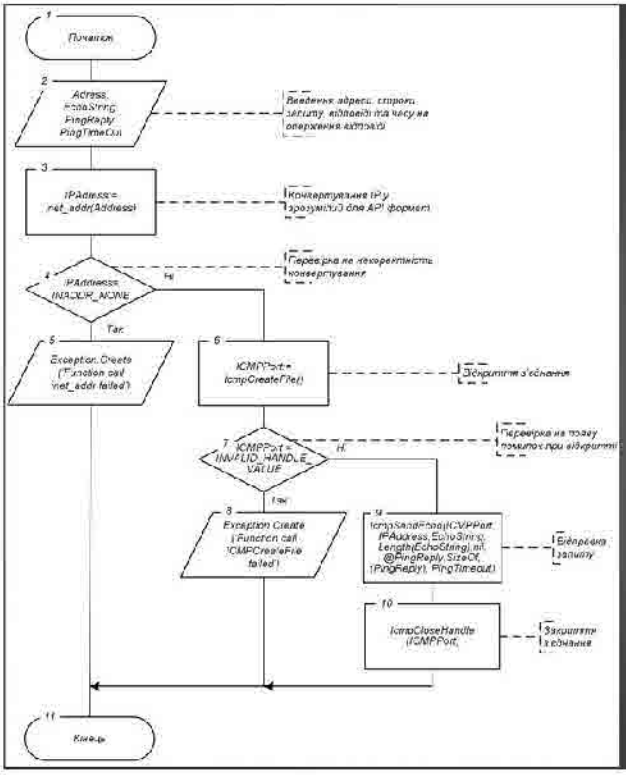

# Розширений алгоритм діагностики мережі



# Алгоритм завдання початкових параметрів для ПЗ діагностики вузлів комп'ютерної мережі

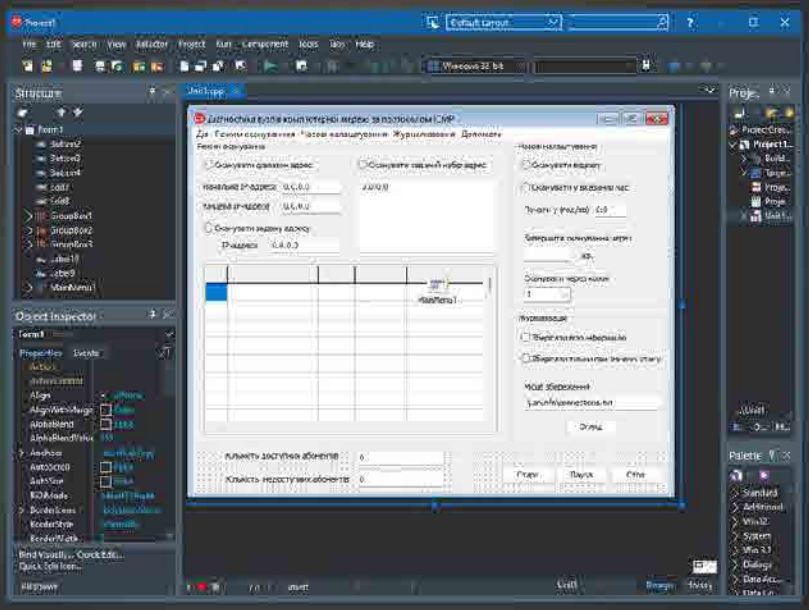


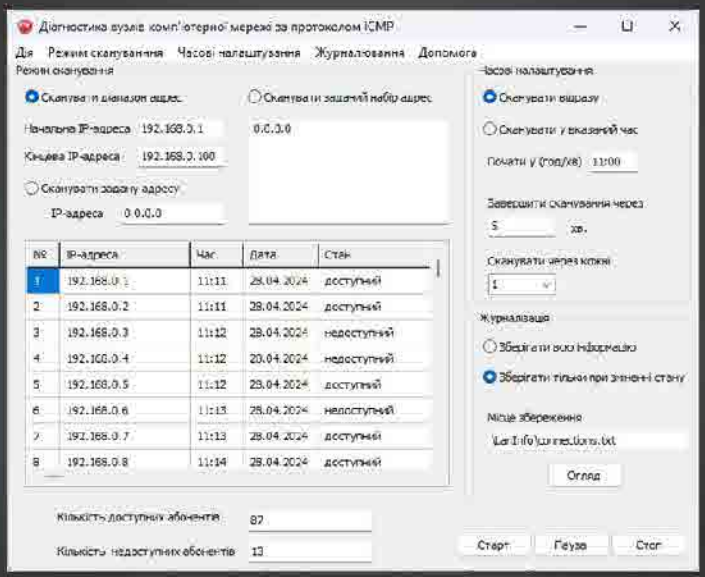




# Алгоритм роботи створеної процедури Ping

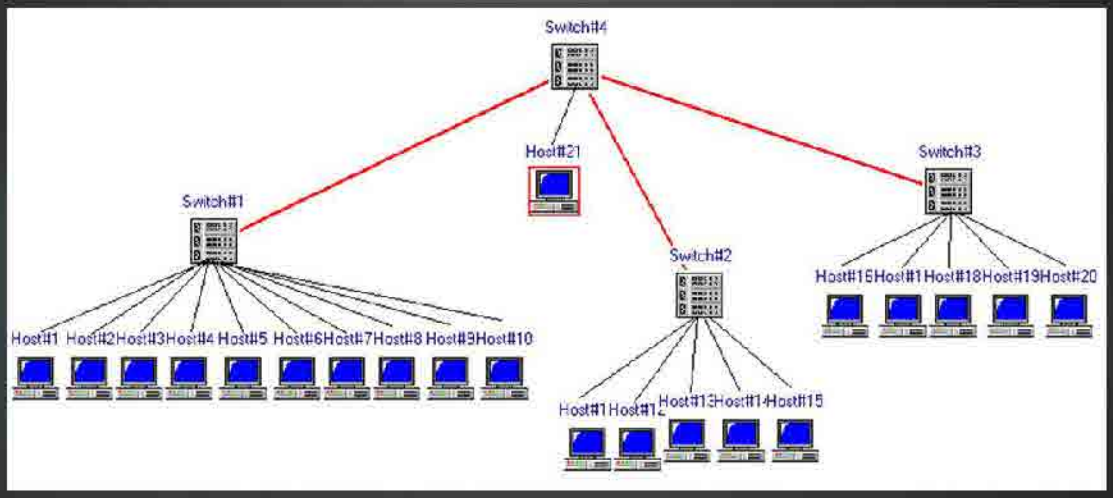
# Середовище розробки Embarcadero RAD Studio C++Builder

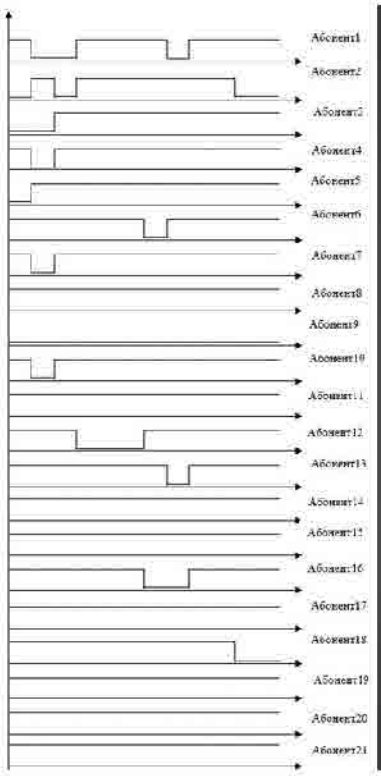


# Головне вікно застосунку діагностики вузлів комп'ютерної мережі



# Побудова мережі для випробування розробленого застосунку




# Файл журналювання стану вузлів комп'ютерної мережі

connections

File Edit View

| | | | | |
|---|---|---|---|---|
| 1 | 192.168.0.1 | 11:11 | 28.04.2024 | доступний |
| 2 | 192.168.0.2 | 11:11 | 28.04.2024 | доступний |
| 3 | 192.168.0.3 | 11:11 | 28.04.2024 | недоступний |
| 4 | 192.168.0.4 | 11:12 | 28.04.2024 | недоступний |
| 5 | 192.168.0.5 | 11:12 | 28.04.2024 | доступний |
| 6 | 192.168.0.6 | 11:12 | 28.04.2024 | недоступний |
| 7 | 192.168.0.7 | 11:12 | 28.04.2024 | доступний |
| 8 | 192.168.0.8 | 11:12 | 28.04.2024 | доступний |
| 9 | 192.168.0.9 | 11:13 | 28.04.2024 | доступний |
| 10 | 192.168.0.10 | 11:13 | 28.04.2024 | недоступний |
| 11 | 192.168.0.11 | 11:13 | 28.04.2024 | доступний |
| 12 | 192.168.0.12 | 11:13 | 28.04.2024 | доступний |
| 13 | 192.168.0.13 | 11:13 | 28.04.2024 | доступний |
| 14 | 192.168.0.14 | 11:13 | 28.04.2024 | доступний |
| 15 | 192.168.0.15 | 11:13 | 28.04.2024 | недоступний |
| 16 | 192.168.0.16 | 11:14 | 28.04.2024 | недоступний |
| 17 | 192.168.0.17 | 11:14 | 28.04.2024 | доступний |
| 18 | 192.168.0.18 | 11:14 | 28.04.2024 | доступний |
| 19 | 192.168.0.19 | 11:14 | 28.04.2024 | недоступний |
| 20 | 192.168.0.20 | 11:14 | 28.04.2024 | недоступний |
| 21 | 192.168.0.21 | 11:15 | 28.04.2024 | доступний |
| 22 | 192.168.0.1 | 11:15 | 28.04.2024 | недоступний |
| 23 | 192.168.0.2 | 11:15 | 28.04.2024 | доступний |
| 24 | 192.168.0.3 | 11:15 | 28.04.2024 | доступний |
| 25 | 192.168.0.4 | 11:15 | 28.04.2024 | доступний |
| 26 | 192.168.0.5 | 11:16 | 28.04.2024 | недоступний |
| 27 | 192.168.0.6 | 11:16 | 28.04.2024 | доступний |
| 28 | 192.168.0.7 | 11:16 | 28.04.2024 | доступний |
| 29 | 192.168.0.8 | 11:16 | 28.04.2024 | доступний |
| 30 | 192.168.0.9 | 11:16 | 28.04.2024 | доступний |
| 31 | 192.168.0.10 | 11:17 | 28.04.2024 | недоступний |
| 32 | 192.168.0.11 | 11:17 | 28.04.2024 | недоступний |

Ln 24, Col 21 2,054 characters 100% Windows (CRLF) UTF-8

# Графіки стану абонентів у мережі

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

# ВІДГУК

керівника на дипломний проект здобувача (здобувачки) освіти

відділення комп'ютерних систем

*Васильєву Вадиму Сергійовичу*

(прізвище, ім`я та по батькові)

Спеціальність: *123 «Комп'ютерна інженерія»*

Освітня програма: *«Обслуговування комп'ютерних систем і мереж»*

Тема дипломного проекту: *Розробка програмних засобів діагностики вузлів комп'ютерної мережі за протоколом ICMP*

## ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ

а) обсяг і якість виконання проекту (графічного матеріалу і розрахунково-пояснювальної записки) *Дипломний проект виконано відповідно технічному завданню. Пояснювальна записка до дипломного проекту містить 84 сторінки. У пояснювальній записці описано реалізацію алгоритму пошуку несправностей мережі, розширений алгоритм діагностики за допомогою процедури ping, створено програмний застосунок для діагностики вузлів ЛОМ за протоколом ICMP. Графічна частина складається з 15 слайдів, оформлених у вигляді презентації, передбачених технічним завданням. Якість виконання пояснювальної записки та слайдів добра.*

б) самостійність роботи над проектом: *Протягом виконання дипломного проекту здобувач освіти Васильєв Вадим поступово та послідовно виконував всі етапи, проявив ініціативу в створенні загальної концепції та реалізації роботи. Всі роботи здобувач освіти виконував самостійно, з оглядом на рекомендації керівника.*

в) теоретична підготовка випускника (випускниці): *Здобувач освіти Васильєв Вадим під час роботи над дипломним проектом вивчив достатньо багато літературних та інтернет-джерел за даною тематикою.*

*Вважаю, що теоретична підготовка дипломника достатня і він готовий до захисту проекту.*

г) вміння розв'язувати виробничі та конструкторські питання *Під час виконання дипломного проекту здобувач освіти Васильєв Вадим показав вміння організовано працювати над поставленим завданням, застосовувати знання у галузі програмування, електроніки, математики, розробляти, встановлювати та налаштовувати спеціалізоване програмне забезпечення, оформлювати слайди та складати презентації, користуючись сучасними комп'ютерними програмними засобами, такими як NetCracker, Embarcadero RAD Studio, Microsoft Visio та ін.*

Оцінка розрахункової частини *Відмінно*

Оцінка графічної частини *Відмінно*

Загальна оцінка *Відмінно*

Прізвище, ім'я, по батькові керівника дипломного проекту *Кривченко Анастасія Анатоліївна*

Місце роботи і посада керівника дипломного проекту *ВСП «Одеський технічний фаховий коледж ОНТУ», викладач спецдисциплін циклової комісії комп'ютерних технологій та програмної інженерії, голова методичної комісії комп'ютерної інженерії Одеської області*

Підпис

«10» 06 2024 р.

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

# РЕЦЕНЗІЯ

на дипломний проект здобувача (здобувачки) освіти

відділення комп'ютерних систем

*Васильєва Вадима Сергійовича*

(прізвище, ім`я та по батькові)

Спеціальність *123 «Комп'ютерна інженерія»*

Освітня програма *«Обслуговування комп'ютерних систем та мереж»*

Керівник дипломного проекту (роботи) *Кривченко Анастасія Анатоліївна*

(прізвище, ім`я та по батькові)

Тема дипломного проекту (роботи) *Розробка програмних засобів діагностики вузлів комп'ютерної мережі за протоколом ICMP*

Обсяг розрахунково-пояснювальної записки *84* сторінок

Обсяг графічної (презентаційної) частини *14* аркушів (слайдів)

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ (РОБОТИ)

а) заключення про ступінь відповідності виконаного дипломного проекту завданню

*Представлений на рецензію дипломний проект відповідає затвердженій темі та виконаний відповідно технічному завданню. Дипломний проект присвячений проблемі діагностики вузлів комп'ютерної мережі за протоколом ICMP та складається з пояснювальної записки, додатку з програмним кодом та мультимедійної презентації, що містить приклади роботи програми.*

б) характеристика виконання кожного розділу дипломного проекту

*Пояснювальна записка складається з основного розділу (аналізу предметної області, проектування, реалізації ПЗ, тестування ПЗ), економічного розділу, розділу охорони праці та додатків. Перелічені розділи поетапно охоплюють розробку, виконані докладно та обґрунтовано. Розділ охорони праці містить загальну інформацію та вимоги до техніки безпеки оператора КТ. Економічний розділ проекту містить розрахунок витрат на НДР та реалізацію проекту.*

в) оцінка якості виконання пояснювальної записки та графічної частини дипломного проекту

*Графічна частина складається з 14 слайдів мультимедійної презентації, виконаної у програмному продукті MS PowerPoint, які містять ілюстративні схеми, скриншоти роботи програмного застосунку, передбачені технічним завданням. Пояснювальна записка виконана акуратно та у відповідності до норм. Якість виконання графічної частини проекту та пояснювальної записки відмінна, розробку виконано у повному обсязі.*

г) перелік позитивних якостей дипломного проекту *Реалізовано програмний засіб діагностики вузлів комп'ютерної мережі за протоколом ICMP.*

*Програмний засіб має дружній графічний інтерфейс, котрий повністю відповідає предметній області.*

д) основні недоліки дипломного проекту

*Було б доцільно представляти результати діагностики у вигляді графіків.*

*В деяких частинах пояснювальної записки присутні незначні помилки оформлення.*

Оцінка розрахункової частини *Відмінно*

Оцінка графічної частини *Відмінно*

Загальна оцінка *Відмінно*

Прізвище, ім'я, по батькові рецензента *Кірєєв Ігор Анатолійович*

Місце роботи і посада рецензента *Державний університет інтелектуальних технологій і зв'язку, доцент каф. інформаційної безпеки та передачі даних*

Підпис:

«14» 06 2024 р.

ПІДПИС ПОСВІДЧУЮ
НАЧАЛЬНИК ВІДДІЛУ
КАДРІВ ДУІТЗ

ВІДДІЛ КАДРІВ

UNICHECK
by Turnitin

Ім'я користувача
**Катерина Григорівна Краснокутська**

ID перевірки
**1016231756**

Дата перевірки
**06.05.2024 18:24:13 EEST**

Тип перевірки
**Doc vs Internet + Library**

Дата звіту
**06.05.2024 18:25:50 EEST**

ID користувача
**100011688**

Назва документа: **4КС-57_Вадим_Васильєв**

Кількість сторінок: **62** Кількість слів: **12022** Кількість символів: **90276** Розмір файлу: **1.47 MB** ID файлу: **1016011917**

## 12.4%
## Схожість

Найбільша схожість: **4.39%** з Інтернет-джерелом

**12.4% Джерела з Інтернету** **343**

Сторінка 64

**Не знайдено джерел з Бібліотеки**

## 0% Цитат

**Вилучення цитат вимкнене**

**Вилучення списку бібліографічних посилань вимкнене**

## 0%
## Вилучень

**Немає вилучених джерел**

## Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті

**Замінені символи** **37**

**ДОЗВІЛ**
**НА РОЗМІЩЕННЯ**
**ВИПУСКНОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ**
**(ДИПЛОМНОГО ПРОЕКТУ)**
**В ЕЛЕКТРОННОМУ РЕПОЗИТАРІЇ ВСП «ОТФК ОНТУ»**

Ми, що нижче підписалися,

***Васильєв Вадим Сергійович,***
здобувач освіти гр. 4КС-57, та

***Кривченко Анастасія Анатоліївна***,
керівник дипломного проекту,

не заперечуємо щодо розміщення електронного варіанту пояснювальної записки до дипломного проекту фахового молодшого бакалавра на тему:

***«Розробка програмних засобів діагностики вузлів комп'ютерної мережі за протоколом ICMP» (автор роботи – Васильєв В.С., керівник роботи – Кривченко А.А.)***

виконаного у ВСП «Одеський технічний фаховий коледж Одеського національного технологічного університету» в 2024 році, у повному обсязі в електронному репозитарії ВСП «ОТФК ОНТУ» для вільного доступу через мережу Інтернет.

Несемо відповідальність за ідентичність електронного та друкованого варіантів випускної кваліфікаційної роботи і даємо згоду на обробку персональних даних.

Виконавець / Васильєв В.С. /

Керівник / Кривченко А.А. /

«10» червня 2024 р.