Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

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

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за другим (магістерським) рівнем вищої освіти
на тему
“ Дослідження та програмна реалізація системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана”

Виконав здобувач вищої освіти
ІІ курсу, групи КІ-25М
ОПП «Комп’ютерна інженерія»
спеціальності F7 «Комп’ютерна інженерія»
_____ Ліподат А.Р.
«____» _____ 2026 р.

Керівник проекту
кандидат технічних наук, доцент
__ _____ Минайленко Р.М.
«____» _____ 2026 р.

Нормоконтроль проекту
_____ Коваленко А.С.
«____» _____ 2026 р.

Рецензент _____
_____

м. Кропивницький

Підписано КЕП
Центральноукраїнський національний технічний університет
Ліподат Антон Романович
Сертифікат: 514B5C86A1E5DA11040000004D8C3A008D76A105
Дійсний з 22.04.2026 19:46:30 по 22.04.2027 19:46:29

Центральноукраїнський національний технічний університет
Факультет *Механіко-технологічний*
Кафедра *Кібербезпеки та програмного забезпечення*
Рівень вищої освіти *магістр*
Галузь знань _F_____*"*Інформаційні технології*"*_____
Спеціальність F7 *"*Комп'ютерна інженерія*"*
Освітньо-професійна (освітньо-наукова) програма _*"*Комп'ютерна інженерія*"*_

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

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ДРУГИМ (МАГІСТЕРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Ліподату Антону Романовичу
(прізвище, ім'я, по батькові)

1. Тема роботи *Дослідження та програмна реалізація системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана*

2. Керівник роботи *Минайленко Роман Миколайович, канд. техн. наук, доцент*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу № 41-13 від 17.08.2026 року

3. Строк подання студентом роботи до захисту *2.12.2026 р.*

4. Мета та завдання випускної кваліфікаційної роботи*: Метою розробки є дослідження та програмна реалізація системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана*

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

1. Призначення та область використання.	*6. Наукова новизна*.
2.Перегляд аналогічних існуючих систем.	*7. Заходи з охорони праці та техніки безпеки..*
3. Опис і обґрунтування проектних рішень.	*8. Висновки.*
4. Етапи програмування системи.	
5. Впровадження системи в промислову експлуатацію	

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

Наукова новизна	*1 аркуш*
Структурна схема системи	*1 аркуш*
Функціональна схема системи	*1 аркуш*
Діаграма процесів	*1 аркуш*
Блок-схема алгоритму роботи додатку	*2 аркуша*

7. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Марченко К.М., к.т.н., доцент	06.10.2026	16.11.2026

8. Дата видачі завдання « 1 » вересня 2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.10.2026 р.	
2.	Постановка задачі, оформлення ТЗ	15.10.2026 р.	
3.	Розробка моделі компонента	20.10.2026 р.	
4.	Розробка структур даних	25.10.2026 р.	
5.	Розробка алгоритмів системи	30.10.2026 р.	
6.	Програмування алгоритмів	10.11.2026 р.	
7.	Розрахунки з охорони праці та техніки безпеки	15.11.2026 р.	
8.	Оформлення ПЗ	17.11.2026 р.	
9.	Попередній захист роботи	2.12.2026 р.	

Дата видачі завдання
« 1 » вересня 2026 р.

Підпис керівника
_____ Минайленко Р.М.
(прізвище та ініціали)
Завдання прийнято до виконання
« 1 » вересня 2026 р.

Підпис здобувача
_____ Ліподат А.Р.
(прізвище та ініціали)

АНОТАЦІЯ

Ліподат А.Р. Дослідження та програмна реалізація системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана. F7 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2026.

В даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначене для системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана.

Метою розробки є створення програмного забезпечення системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана.

Об'єктом дослідження є системи розподілу мережевого трафіка.

Предметом дослідження є методи створення програмного забезпечення системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана.

Методи дослідження базуються на методах теорії побудови комп'ютерних мереж, методах математичної статистики, методах розробки програмного забезпечення.

Результат роботи - програмна реалізація системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана.

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

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

Програма реалізована на мові високого рівня Delphi.

Ключові слова: програмне забезпечення, система, мережевий трафік.

ABSTRACT

Lipodat A.R. Research and Software Implementation of a Network Traffic Distribution System Based on an Improved Ford-Bellman Algorithm. F7 Computer Engineering. Central Ukrainian National Technical University. Kropyvnytskyi. 2026.

This final qualification thesis, submitted for the second (master's) level of higher education, presents the development of software designed for a network traffic distribution system based on an enhanced Ford-Bellman algorithm.

The objective of this development is to create software for a network traffic distribution system based on an enhanced Ford-Bellman algorithm.

The object of the study is network traffic distribution systems.

The subject of the study consists of methods for developing software for a network traffic distribution system based on an enhanced Ford-Bellman algorithm.

The research methodology is based on principles from computer network theory, mathematical statistics, and software engineering.

The result of the work is a software implementation of a network traffic distribution system based on an enhanced Ford-Bellman algorithm.

During the development of the software model, an analysis of existing hardware and software tools was conducted. All components of the developed software are described in detail.

A user-friendly interface has been developed, and instructions for using the software are provided.

The program is implemented using the high-level language Delphi.

Keywords: software, system, network traffic.

ВІДОМІСТЬ ПРОЕКТУ

		Текстові документи			
1	А4	Пояснювальна записка	81	–	
2	А4	Робоча програма	47	–	Додаток А
		Графічні документи			
3	А4	Наукова новизна	1	–	
4	А4	Структурна схема	1	–	
5	А4	Функціональна схема роботи системи	1	–	
6	А4	Діаграма взаємодії процесів	1	–	
7	А4	Блок-схема основної програми	1	–	
8	А4	Блок-схема роботи підпрограми	1	–	

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ.....9
ВСТУП..... 10
1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ 13
1.1 Призначення системи..... 13
1.2 Область застосування 15
2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ 16
2.1 Огляд існуючих систем, технологій, архітектур та програмних рішень за профілем теми випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти..... 16
2.2 Обґрунтування вибору засобів для побудови системи та мови програмування..... .18
2.3 Розгорнута постановка завдання 29
3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ 30
3.1 Опис функціонування системи 30
3.2 Розробка структурної схеми..... 32
3.3 Розробка функціональної схеми 36
3.4 Розробка діаграми процесів 40
4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ 43
4.1 Розробка блок-схем та опис алгоритмів функціонування системи 43
5 ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ..... 57
6 НАУКОВА НОВИЗНА 65
7 ЗАХОДИ З ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ 66
7.1 Вступ..... 66
7.2 Аналіз умов праці..... 68
7.3 Техніка безпеки та протипожежна профілактика..... 71
7.4 Розробка заходів з охорони праці..... 73
7.5 Висновки 74
8 ОСНОВНІ ВИСНОВКИ 75

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 76

ПЕРЕЛІК СКОРОЧЕНЬ, СИМВОЛІВ І СПЕЦІАЛЬНИХ ТЕРМІНІВ

Delphi - мова програмування;

Java – мова програмування;

JavaScript – мова програмування;

Pascal – мова програмування;

Windows – операційна система;

КТД – комплекс технічної документації;

КТЗ – комплекс технічних засобів;

ПЗ – програмне забезпечення;

ПЗНП – програмні засоби навчального призначення;

ПЕОМ– персональна електронно обчислювальна машина;

ПК – персональний комп'ютер;

ТД – технічна документація;

ОС – операційна система.

ВСТУП

Актуальність теми. У зв'язку з розширенням глобальної мережі Internet зростає необхідність впровадження нових оптимізаційних алгоритмів, пов'язаних зі швидкістю обміну даних між комп'ютерами в єдиній мережі.

Комп'ютерні мережі швидко завойовують світ. Комп’ютерні системи перетворюються у величезні сховища даних, доступні з будь якої місця у світі. Будь-яка сучасна фірма, будь-який офіс оснащений мережею де, сотні тисяч людей працюють на персональних комп'ютерах, приносячи реальну користь спільноті.

В основному для роботи в Internet використовуються браузери – програми, що дозволяють проводити обмін текстовою, графічною та звуковою інформацією, використовуючи популярну, просту у використанні, мультимедійну службу Інтернет.

Мета й завдання дослідження. Метою випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти є створення програмного забезпечення системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана. Для досягнення поставленого завдання необхідно:

- провести аналіз основних існуючих корпоративних мереж;
- проаналізувати існуючі браузери;
- розробити алгоритми визначення оптимального маршруту передачі інформації по глобальній мережі;
- створити математичну модель мережевого браузера і корпоративного середовища;
- створити програмне забезпечення системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана.

Об’єктом дослідження є системи розподілу мережевого трафіка.

Предметом дослідження є методи створення програмного забезпечення системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана.

Методи дослідження базуються на методах теорії побудови комп'ютерних мереж, методах математичної статистики, методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених метою дослідження, отримані наступні результати:

– Удосконалено систему розподілу мережевого трафіка.

– Розроблено вітчизняний продукт - систему розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана, який має більш широкі можливості, на відміну від існуючих аналогів.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно реалізувати систему розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана.

1. Розроблено модель корпоративного середовища і мережевого браузера;

2. Створено систему розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана.

3. Реалізовано відповідне програмне забезпечення - мережевий браузер з можливістю доступу як до WWW- протоколу, так і до сервісу FTP, поштового сервісу SMTP, а також можливістю обміну інформацією між двома комп'ютерами в реальному режиміас і математична модель корпоративної мережі, для імітації передачі інформації в глобальному середовищі, в якій реалізований розроблений вдосконалений алгоритм Форда-Белмана для пошуку оптимального шляху між маршрутизаторами.

Достовірність наукових результатів підтверджена теоретичними викладеннями, даними комп'ютерного моделювання, коректними дослідженнями параметрів на функціонуючій обчислювальній мережі, а також відповідністю отриманих результатів окремим результатам, наведеним у науковій літературі.

Основні положення випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти надруковані у статті збірника праць молодих науковців ЦНТУ, випуск №17

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ЗАСТОСУВАННЯ СИСТЕМИ

1.1 Призначення системи

Розроблене у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти програмне забезпечення (ПЗ) призначене створення системи розподілу мережевого трафіка з використанням вдосконаленого алгоритму Форда-Белмана.

Під час завантаження ПЗ на екрані монітора ЕОМ відображується функціональна схема комп’ютерної мережі. Після цього при натисканні клавіші ENTER, програма переходить в текстовий режим з UNIX-подібним інтерфейсом і формує запит у користувача адреси одержувача, та дані типу "String".

Послідуючим кроком є те, що кожен комп'ютер (або маршрутизатор), по якому проходить пакет даних, виводить на екран ЕОМ повідомлення про приймання та подальше відправлення пакета адресату і інтервал часу, за який він отримує і надсилає пакет інформації. Оптимальний маршрут для передачі інформаційних пакетів розраховується з використанням вдосконаленого алгоритму Форда-Беллмана

Типи змінних і даних головної програми представлено нижче:

```
const AdjacencyMatrix: array [1..VertexQuantity, 1..VertexQuantity] of byte =
                (0,1,0,1,0,0,0),
                (1,1,1,0,1,0,1),
                (0,1,0,1,0,0,0),
                (1,0,1,0,1,0,0),
                (0,1,0,0,1,1,0),
                (0,0,0,0,1,0,1),
                (0,1,0,0,0,1,0)) - матриця суміжності маршрутизаторів;
TYPE TAddr = record
 router: byte;
 domain: byte;
```

comp: byte;

END - адреса комп'ютера, яка складається з номера маршрутизатора, номера області конкретного маршрутизатора і номера конкретного комп'ютера;

TYPE TBatch = record

from: TAddr;

to_: TAddr;

data: string;

path: array [1..20] of byte; {path is chain of router numbers}

END - пакет, що складається з адреси відправника, адреси одержувача, даних і шляху проходження пакета;

TYPE TComp = object - модель комп'ютера, яка складається з адреси, комірки пам'яті для одержання або пересилання пакета;

addr: TAddr;

mem: TBatch;

Procedure Send2Router (batch: TBatch) - процедура надсилання пакета до маршрутизатора;

Procedure Send (batch: TBatch) - процедура надсилання пакета в середині мережі;

Procedure Receive (batch: TBatch; byRouter: boolean) - приймання пакета;

END;

TYPE TRouter = object - модель маршрутизатора, яка складається з номера маршрутизатора, координат, і комірки пам'яті;

num: byte;

x, y: integer;

memory: Tbatch;

state: boolean;

VAR computers: array [1..38] of TComp – певний масив комп'ютерів глобальної мережі;

routers: array [1..7] of TRouter - масив маршрутизаторів;

OptimalPath: array [1..49] of byte – самий оптимальний шлях, розрахований маршрутизатором;

Procedure Receive (routerNum: byte; batch: TBatch) - приймання пакета;

Procedure Send2Comp (batch: TBatch) - відправка пакету у своїй мережі;

Procedure CalcMinPath (sender, target: byte) - обчислення оптимального шляху відправки;

Procedure Send2NextRouter (batch: TBatch; currentRouter: byte) - відправка на наступний маршрутизатор;

END;

1.2 Область застосування

Розроблене в даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти програмне забезпечення дозволяє використовувати його в системі розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда - Белмана і має наступні можливості:

- завантаження Web-сторінок;
- операції з відправлення та одержання пошти;
- обмін файлами за протоколом FTP;
- можливість обміну символьної інформацією між комп'ютерами, на яких повинно бути встановлене відповідне програмне забезпечення.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи системі розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

2. ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

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

Комп'ютерна мережа (КМ) складається з елементів, серед яких вирізняють комп'ютери, що надають ресурси в мережі (сервери), комп'ютери, що забезпечують доступ до ресурсів серверів (клієнти) мережі, середовище (media), в якому створені з'єднання, ну і, власне, самі ресурси, тобто файли, процесори, принтери, і т.д.. Принципову можливість спільного використання пристроїв і даних забезпечує комп'ютерна мережа.

З'єднання комп'ютерів в мережі може здійснюватися різними способами і, власне, різним типовим способам надаються різні власні назви.

Розділяють комп'ютерні мережі :

- з виділеними серверами;
- однорангові комп'ютерні мережі.

На теперішній час найбільше поширення отримали комп'ютерні мережі яким притаманна архітектура клієнт-сервер. Такі КМ використовують центральний сервер для обслуговування запитів клієнтів, а однорангові мережі дозволяють будь-якій робочій станції функціонувати одночасно в якості сервера, якщо це прописано у поставленому завданні.

У КМ з архітектурою клієнт-сервер спеціалізований комп'ютер (чи виділений сервер) використовується з метою встановлення усіх розподілених ресурсів. Саме таке рішення пришвидшує доступ користувачів до централізованих ресурсів мережі та має певні особливості, а саме:

- адміністрування мереж значно спрощується за рахунок відносно невеликої кількості серверів, що містяться у мережі і їх вузькопрофільну спеціалізацію;

- наявні високі вимоги до виділеного сервера - для створення високої продуктивності сервера має бути присутнім великий об'єм оперативної пам'яті ОЗП), а також диск великої ємності і. власне. процесор з відповідно великою продуктивністю;

- при виникненні збоїв чи помилок у роботі сервера комп'ютерна мережа буде недієспроможною.

При відсутності виділеного сервера в одноранговій мережі, всі комп'ютери мають рівні права і можливості і можуть розглядатися як сервери, і як клієнти. Частіше всього на практиці однорангові мережі містять до десяти ЕОМ.

Однорангова мережа на основі сервера містить виділений сервер і може містити як один, так і декілька серверів. Кожен сервер при цьому може мати певне призначення:

- файл-сервери;

- принт-сервери;

- сервери додатків, на яких виконуються прикладні задачі;

- поштові сервери;

- факс-сервери;

- сервери комунікаційні, які керують потоком інформаційних даних і повідомленнями пошти між мережею, якій вони належать, та іншими мережами, мейнфреймами (великими ЕОМ) або віддаленими користувачами через модеми та телефонні лінії;

- сервери служб каталогів, які забезпечують пошук, захист та зберігання інформаційних даних у мережі.

У комбінованих мережах зібрані найкращі властивості однорангових КМ і КМ створених на основі серверів. Використовується сервер, але є й інші окремі компоненти з функцією дозволу доступу до своїх даних.

Створені базові топології локальних мереж (ЛМ) - є основними видами конфігурацій з'єднань елементів мереж за допомогою дротів.

Проведемо аналіз трьох базових топологій, а саме:

- шина;
- зірка;
- кільце.

Шина (або лінійна шина) - це топологія, що показана на рис. 2.1:

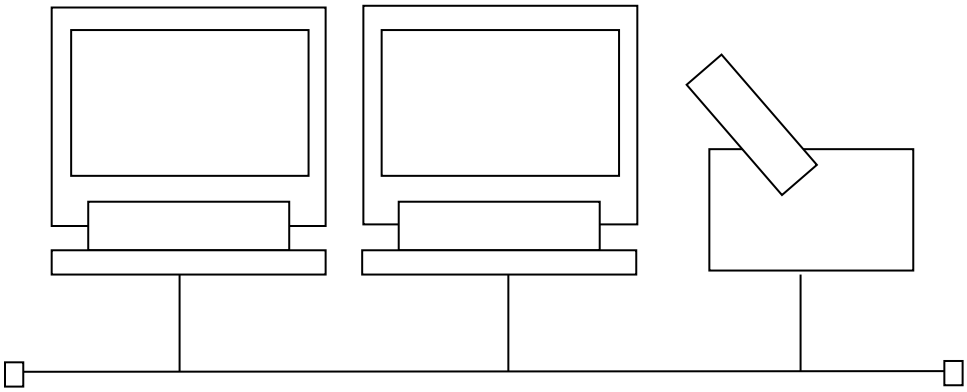


Рисунок 2.1 – Лінійна шина

На сьогоднішній день (якщо не брати до уваги анкет, заповнених користувачами) основна частина інформаційного трафіку передається в одному напрямку. а браузери зчитують файли з Web-сервера. Таке становище змінилось при використанні HTTP 1.1 який дозволяє записувати файли на Web-сервер. Метод put використовується програмістами сторінок Web, але ін може стати в нагоді і звичайним користувачам для зворотного зв'язку з інформаційними вузлами.

Шина топологія є пасивною топологія. Комп'ютери тільки проводять обробку. і не регенерують сигнали. Підключення кабелю здійснюється за допомогою Баррел-конекторів і репітерів.

Баррел-конектори - це спеціальні металеві сполучники які дозволяють зрощувати дротові кабелі, але при великій кількості сполучень сигнал має тенденцію відчутно загасати. Для вирішення даної проблеми (збереження фізичних параметрів сигналів, що поширюються в комп'ютерних мережах) застосовують спеціальні пристрої - репітери.

Репітер - це повторювач-формувач, який підсилює корисний сигнал.

В топологія «зірка» передбачено підключення всіх комп'ютерів за допомогою кабельних сегментів до центрального вузла. При чому розрізняють два види цієї топології - пасивна зірка, в основі якої відсутній ЕОМ-абонент, а кабелі з'єднані при допомозі концентратора (hub) і активна зірка, в основі якої є ЕОМ, з допомогою якої відбувається обмін інформації в мережі.

Концентратор (hub) – це пристрій, що служить для об'єднання декількох сегментів мережі і не має можливості перетворювати передану інформацію. Сигнали від передаючої ЕОМ надходять через концентратор до всіх інших ЕОМ які, входять в мережу. Концентратори бувають:

- активні;
- пасивні;
- гібридні.

Топологія активна зірка забезпечує:

- безконфліктне управління, але порушення в роботі центру призводять до виходу з ладу всієї мережі;

- мережа з такою топологією мало чутлива до виходу з ладу ділянок з'єднувального кабелю.

Топологія кільце передбачає передачу сигналів по кільцю в одному напрямку, так, що сигнали проходять через кожен комп'ютер (рис.2.2).

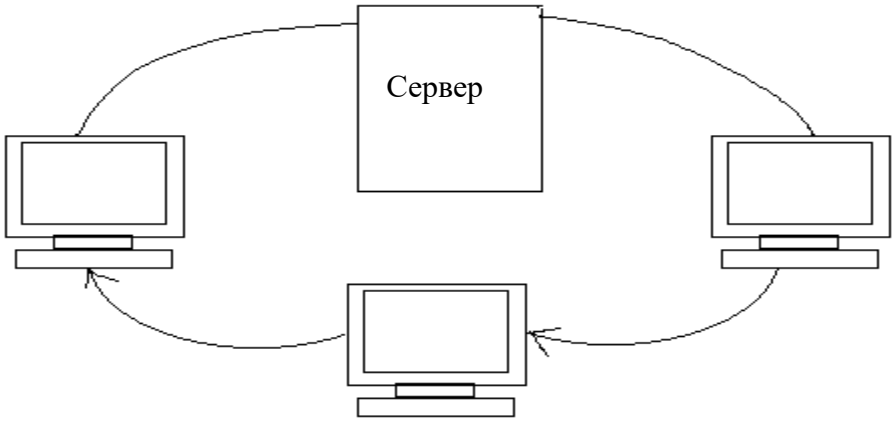


Рисунок 2.2 - Топологія "Кільце"

На відміну від пасивної топології "шина", тут кожній ЕОМ властива функція репітера, тобто сигнал підсилюється і передається наступному комп'ютеру.

Приймаються дані з шини комп'ютера і перетворюються в послідовний бітовий код, який використовується при передачі по кабельний лініям. Плата адаптера мережі повинна вказувати своє місцезнаходження або адресу в мережі, щоб її можна було виокремити серед від інших плат. Мережеві адреси визначено комітетом IEEE, який закріпив за кожним виробником мережевого обладнання визначені інтервали адрес. Ці адреси «прошиті» в мікросхеми і завдяки цьому кожна плата і кожна ЕОМ мають власну унікальну адресу в мережі. Плата мережевого адаптера запитує по шині у комп'ютера дані з пам'яті, і якщо вони надходять швидше, ніж їх можна передати плата, то дані на деякий час розміщуються в буфері.

На початку, перед передачею даних, передаюча плата генерує обмін інформацією з приймаючою платою, здійснюючи певні узгоджувальні дії про параметри передачі:

- Розмір блоку переданих даних.

- Об'єм даних, переданих без звіту про їх отримання.

- Час між передачами блоків даних.

- Час, протягом якого необхідно надіслати підтвердження про отримання даних.

- Обсяг даних, які спроможна прийняти кожна плата без переповнення.

- Швидкість передачі даних.

Такі протоколи це набір правил і процедур, що призначені для встановлення зв'язків в комп'ютерній мережі.

Набір протоколів мережі Internet, TCP/IP (Transmit Control Protocol/Internet Protocol) – це визначений промисловий стандарт протоколів забезпечуючих зв'язок між різними типами комп'ютерів.

Розглянемо декілька протоколів передачі даних високого рівня. саме:

- WWW (World Wide Web) – є найважливішим ресурсом мережі Internet, забезпечуючим гіпертекстове представлення інформації та гіпертекстові посилання для єдиного доступу у глобальній всесвітній мережі.

- Gopher – це удосконалена система передачі інформаційних файлів, яка дозволяє, з допомогою системи меню, переглядати певні списки ресурсів і пересилати потрібні матеріали та переглядати певні дані. ЕОМ, які використовують Gopher, можна назвати з'єднаними в єдину пошуково-інформаційно систему GopherSpace.

- WAIS – є індексною пошуковою системою глобальних інформаційних серверів (сервера).

- SNMP (Simple Network Management Protocol) – це протокол мережі Internet для моніторингу мережевих компонентів і ,власне, самої мережі.

- File Transfer Protocol – це протокол Internet-мережі для передачі інформаційних файлів. Він підтримує процес передачі файлів між віддаленими і локальними комп'ютерами. Крім того, містить команди, які реалізують реверсивну передачу даних і ASCII-файлів між ЕОМ.

- Telnet – це протокол Internet-мережі який застосовується з метою обробки даних та реєстрації на віддалених хостах.

- SMTP (Simple Mail Transfer Protocol) – це протокол мережі Internet для електронної пошти.

- TCP (Transmission Control Protocol) - TCP/IP – це протокол для доставки даних (гарантованої), які складаються з послідовності фрагментів.

- IP (Internet Protocol) – це протокол мережевого рівня з наборів Internet-протоколів, для забезпечення правильної передачі пакетів і є базовим елементом Internet-технології. Головна його частина – це таблиця маршрутів, яка використовується під час прийняття рішень про маршрутизацію IP-пакетів.

- OSPF (Open Shortest Path First) – це алгоритм маршрутизації, де використовується стан каналів (основою його є алгоритм Дейкстри). Даний алгоритм проводить обчислення маршрутів з урахуванням кількості

маршрутизаторів, через які пройде пакет інформації в напрямку до одержувача, а також враховує пропускну спроможність лінії мережі, трафік і вартість.

- Набір протоколів IPX/SPX та NWlink (Internetwork Packet Exchange/Sequenced Packet Exchange) – це стек протоколів, які використовуються в мережі Novell. Даний протокол відносно швидкий і невеликий, з можливістю підтримки маршрутизації. NWlink – реалізований компанією Microsoft і використовується в якості транспортного протоколу маршрутизації.

Розглянемо коротко основні види симуляторів для створення комп'ютерних мереж.

Екосистема SDN-контролерів ONOS та OpenDaylight.

ONOS Web UI: Має мінімалістичний світлий або темний веб-інтерфейс. Лінки (лінії графа) динамічно змінюють товщину або колір залежно від поточного завантаження в Мбіт/с. У правій бічній панелі (Summary) в реальному часі виводяться лічильники пакетів, кількість активних пристроїв та знайдені алгоритмом оптимальні шляхи (Intents).

OpenDaylight (ODL) DLUX: Поєднує графічний граф мережі та деревоподібне меню конфігурації (Yang UI). Через цей інтерфейс можна відкрити детальні властивості кожного ребра, де відображається системний опис лінку (затримка, дроп пакетів), що дозволяє візуально контролювати, чому алгоритм Форда-Белмана вибрав саме цей маршрут.

Приклади інтерфейсів ONOS та OpenDaylight показано на рис. 2.3. і рис. 2.4 відповідно:

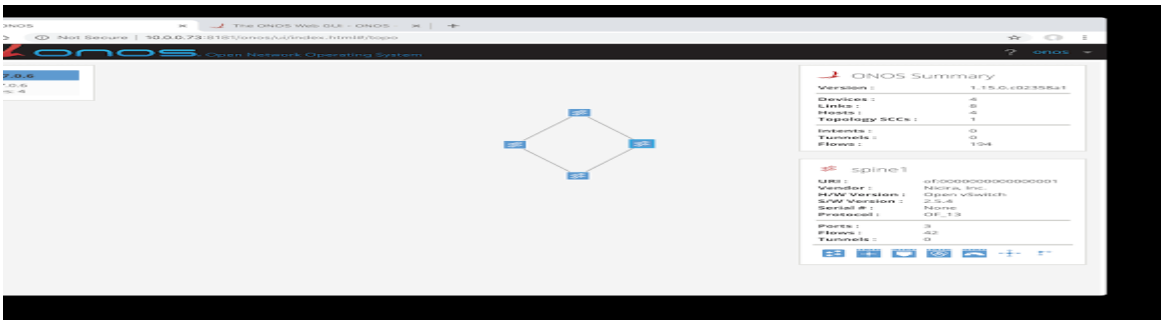


Рисунок 2.3 - Приклад інтерфейсу ONOS

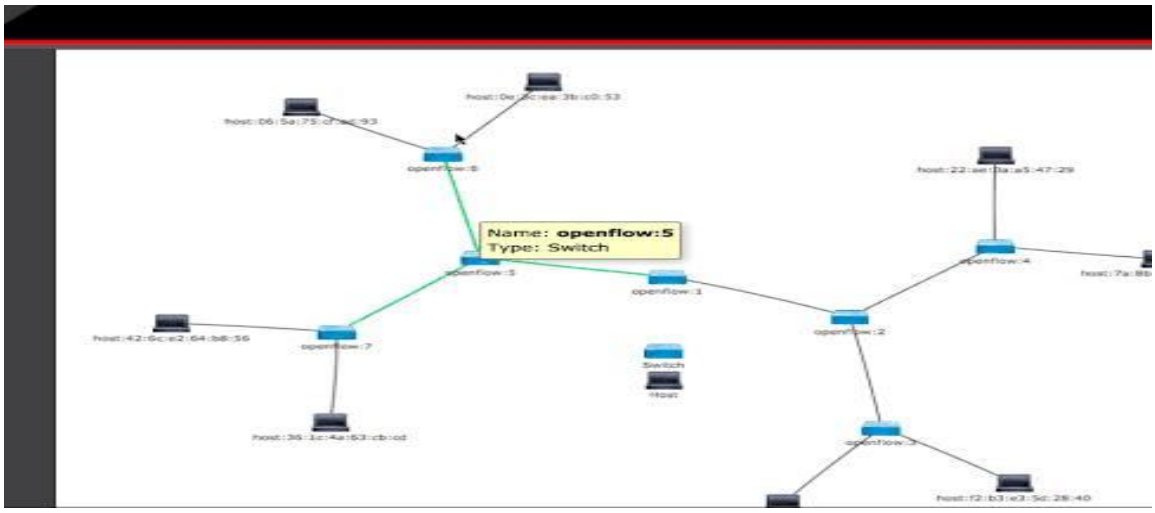


Рисунок 2.4 - Приклад інтерфейсу OpenDaylight

Симулятори мереж для тестування (Mininet/MiniEdit). MiniEdit: Це класичний візуальний редактор (побудований на Python + Tkinter). Ліворуч розміщено панель інструментів (Хост, Комутатор, Контролер, Лінк). Клікнувши правою кнопкою миші на лінк, розробник відкриває діалогове вікно «Link Properties», де можна штучно задати метрики: смугу пропускання (Bandwidth у Mbps), затримку (Delay у ms) та відсоток втрат пакетів (Loss у %). Після запуску симуляції скрипт розраховує маршрути згідно з логікою вашого коду.

Приклади інтерфейсів Mininet/MiniEdit показано на рис. 2.5. і рис. 2.6 відповідно:

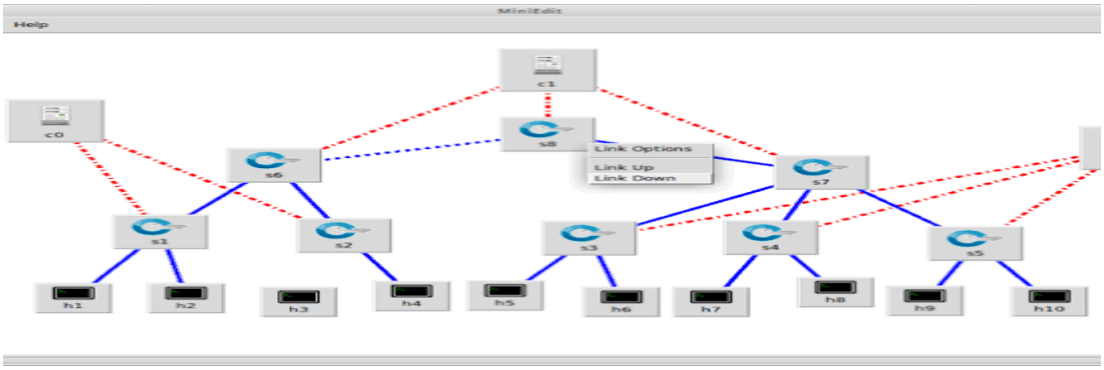


Рисунок 2.5 - Приклад інтерфейсу Mininet

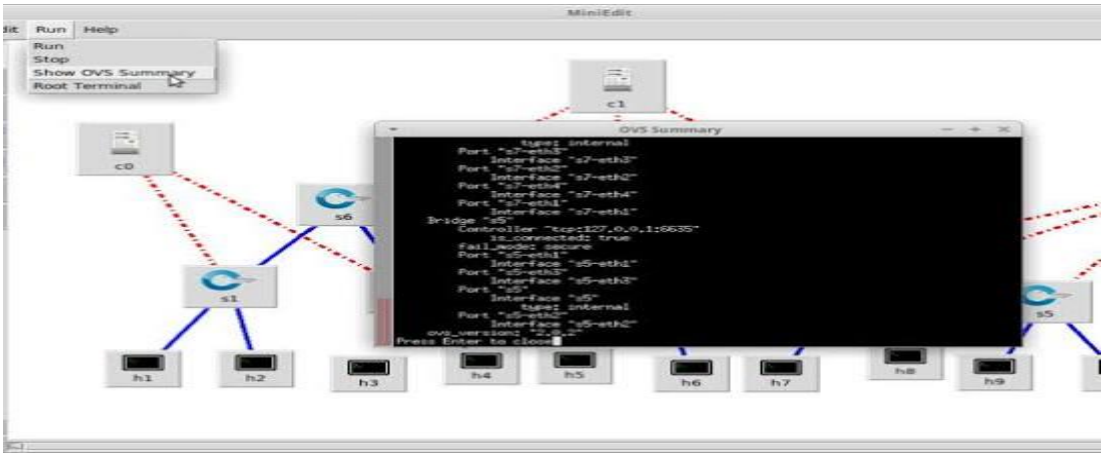
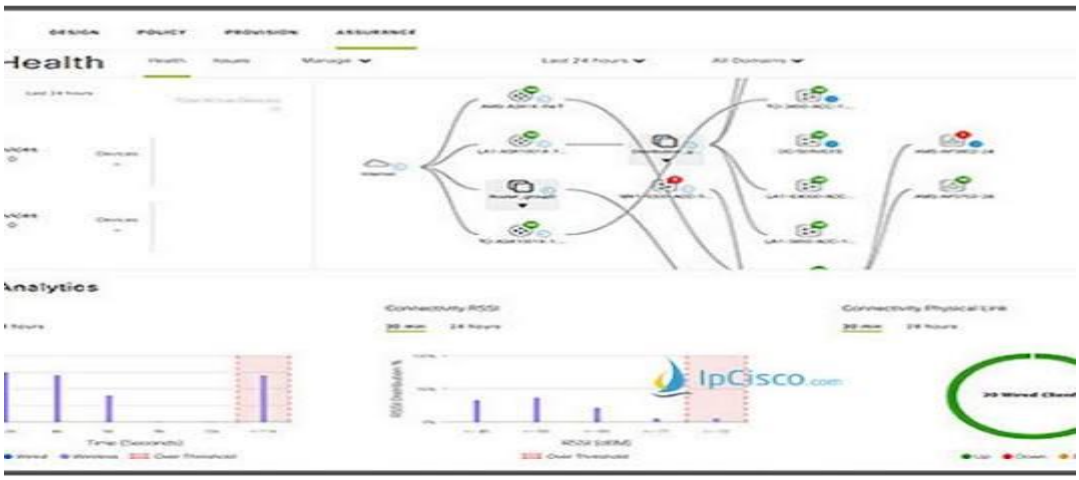


Рисунок 2.6 - Приклад інтерфейсу MiniEdit

Промислові аналітичні панелі (Cisco DNA Center).

Cisco DNA Center Dashboard: Головний екран розбитий на інформаційні блоки (віджети). Інтерфейс містить секторні діаграми (Pie Charts), лінійні графіки завантаження каналів у часі та KPI-метрики якості обслуговування (QoS). Окремий екран «Network Health» показує глобальну карту мережі (можна накласти на реальну географічну карту) із кольоровими індикаторами стабільності кожного вузла.

Приклади інтерфейсів Cisco DNA Center Dashboard показано на рис. 2.7:



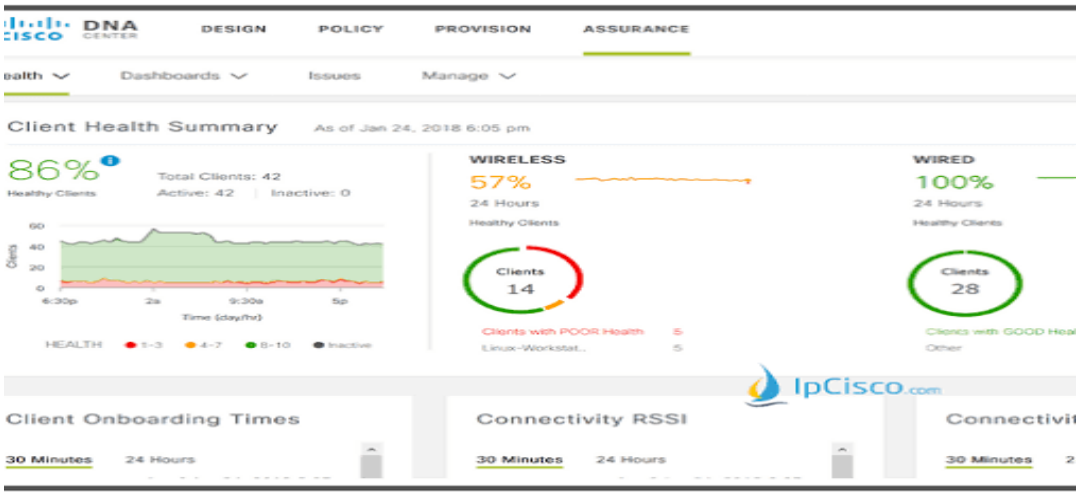


Рисунок 2.7 - Приклад інтерфейсів Cisco DNA Center Dashboard

Тобто на основі проведеного огляду існуючих систем та рішень імітаційні системи розподілу мережевого трафіка існують, але в них використовується звичайний алгоритм Форда-Белмана.

2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

Для реалізації ПЗ було обрано інтегроване середовище розробки Delphi (Integrated Development Environment) – IDE, яке включає в себе кілька основних елементів:

- Піктограми (Speed Buttons) - піктографічні кнопки, дублюючі команди меню. Найменування кожної піктограми можна отримати безпосередньо на екрані в маленькому вікні контекстної підказки, яка з'являється при затриманні покажчика миші на деякий час на зображенні відповідної піктограми.

- Головне меню (Menu Bar) - стандартне меню в стилі Windows. Біля більшості команд меню в Delphi зображені піктограми. Це ті ж самі піктограми, як і на панелі інструментів.

- Палітра компонентів (Component Palette) - тут представлені піктограми компонентів, які входять в бібліотеку VCL. Отримати інформацію про ім'я компонента, який знаходиться за тією або іншою піктограмою, можна тим же способом, який застосовується для піктограм, що є на панелі інструментів.

- Категорії палітри – у кожний момент часу в полі палітри відкрита тільки одна категорія компонентів.

- Вкладки властивостей і подій - клікнувши мишею на одній з двох вкладок діалогового вікна інспектора об'єктів Object Inspector, відкриється або вкладка властивостей (Properties) об'єктів компонентів, включених в екранну форму, або вкладка подій (Events) об'єктів компонентів. Властивості описують атрибути об'єкта - розмір кнопки, шрифт текстової етикетки. Події ж є деякими діями (наприклад, клацання на кнопці або натискання клавіші).

- Інспектор об'єктів (Object Inspector) - вікно відображає списки усіх подій і властивостей одного або іншої кількості об'єктів компонентів, що обрані в екранній формі.

- Вікно проектування екранної форми (Form window) - при розробці значної кількості програм екранна форма є деяким візуальним представленням головного вікна програми. Точкова сітка на полі форми проектування допомагає точніше вибрати місце встановлення.

- Діалогове вікно Module Explorer - засіб, який включено у версію мови Delphi 4. Вікно Module Explorer використовується для виведення інформації про використовуваний модуль програми: перелік класів, список інших модулів, які він використовує, змінних, методів, об'єктів і т.і.

- Редактор коду програми (Code editor window) – у цьому вікні є можливість для переглядання та редагування текстів програм на мові Паскаль, пов'язаної з будь-якою екранною формою додатка, який має розроблюватись. Більшість загальних фрагментів програмного коду (оголошення і заготовки процедур обробки подій) Delphi вносить автоматично в програму. Потім обробник наповнює заготовку змістом - включає необхідні оператори визначі

поведінку програми в залежності від тої або іншої ситуації (наприклад, клацання мишею або вибір меню команди).

Редакція Client-Server Delphi надсилається з повним набором засобів підтримки та створення додатків для роботи з Internet. Ці компоненти є досить корисними не тільки для отримання доступу до різноманітних служб і ресурсів Internet, але і для продукування засобів ПЗ для віддаленого доступу до баз даних. Для прикладу, фірма Inprise використовувала компоненти Delphi під час розробки додатків, які мали змогу накопичувати інформацію про програмні баги.

Середовище Delphi пропонує різноманітні набори компонентів для програмування в Internet. Ці складові не мають візуальних зображень в часі прогону і з'являються як піктограми у віконій формі. Всі компоненти - це елементи керування ActiveX, вбудовані в пакувальники компонентів Delphi. Нижче наведені компоненти управління Internet ActiveX в порядку їх розташування на екрані:

- FTP (Internet File Transfer Protocol) – для пересилання файлів і даних між клієнтом і локальним комп'ютером.

- HTML (Hypertext Markup Language) – для підтримки перегляду документів формату HTML (Також може використовуватися для синтаксичного аналізу команд HTML).

- HTTP (Hypretext Transport Protocol) - витягує файли документів у форматі HTTP без можливості обробки зображень і попереднього перегляду.

- NNTP (Network News Transport Protocol) - підтримує відправку новин і читання інформації.

- POP (Post Office Protocol) - підтримує поштову службу UNIX або інших серверів зі стандартом POP3.

- SMTP (Simple Mail Transfer Protocol). – використовується для забезпечення доступ до поштових серверів SMTP.

- TCP (Transmission Control Protocol) - підтримує протокол TCP для класу додатків клієнт/сервер.

Для формування засобів доступу до HTML-документів через Internet достатньо розмістити компонент HTML в екранну форму і створити процедуру обробки дії, яка буде викликати метод RequestDoc такого компонента, як показано нижче:

HTML1.RequestDoc (URLs.text);

Передбачається, що параметр URLs є об'єктом ComboBox чи якимсь іншим компонентом, з властивістю text. Із визначеного діапазону об'єктів параметру URL може бути вибранийний будь-який об'єкт. Компоненту RequestDoc можна надати і параметр-рядок:

```
HTML1.RequestDoc ('www.inprise.com');
```

Об'єкти компонента HTML досить корисні при використанні для отримання доступу до вихідних текстових даних HTML-документів. Для цього необхідно зв'язатися з вузлом, шляхом виклику методу компонента RequestDoc. Результатом чого буде отримано доступ до властивостей SourceText HTML-документа, який є порядковим списком з необробленим текстовим документом. Пропонований разом з Delphi демонстраційний Web-браузер використовує властивості SourceText для того, щоб продемонструвати у вікні вихідний HTML-документ, з додаванням тексту у вікно редактора Memo. І можливим є використання такого виразу:

```
Memo1.Lines.Clear;
Memo1.Lines.Add (HTML1.SourceText);
```

Існує дві події компонента HTML, які є корисними для визначення моменту початку і закінчення пошуку та вибирання документа. Спочатку маємо вибрати об'єкт HTML і клацнути на вкладці Events вікна Object Inspector. Потім потрібно двічі клацнути в рядках подій OnBeginRetrieval і OnEndRetrieval, для того, щоб Delphi сформувала заготовки процедур для оброблення кожного з них.

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускну кваліфікаційну роботу за другим (магістерським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана..

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

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

б) вибрати та обґрунтувати методику розробки системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана. А також розробити функціональну та структурну схеми системи;

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

г) організувати інтерфейс користувача;

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

е) розробити заходи по охороні праці при впровадженні та експлуатації системи

ж) сформувати висновки про виконаний обсяг робіт та одержані результати.

3. ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

В загальному значенні слово маршрутизація означає пересилання інформації від джерела інформації до пункту призначення через об'єднану мережу. При цьому, як правило, на шляху пересилання зустрічається принаймні один вузол. Маршрутизації доволі часто протиставляється об'єднання мережі за допомогою мосту, яке, в довільному розумінні цього способу, виконує ті ж самі функції.

Головною відмінністю між ними є те., що об'єднання за допомогою моста має місце на другому рівні ISO (еталонної моделі), у той час як маршрутизація зустрічається на третьому рівні. Ця різниця і пояснюється те, що об'єднання за мостовою схемою і маршрутизація використовують різні типи інформації в процесі переміщення її (від джерела до місця призначення). В результаті чого об'єднання за допомогою моста і маршрутизація виконують свої певні задачі різними способами. На практиці існує кілька різних видів об'єднання за допомогою мостів та маршрутизації.

Більше двох десятиліть тема маршрутизації висвітлювалася в науковій комп'ютерній літературі, однак з точки зору комерції маршрутизація стала набувати популярності тільки у 70-х роках минулого століття. На протязі цього періоду мережі були доволі простими, а об'єднання великомасштабних мереж стало впроваджуватись тільки останнім часом.

Маршрутизація складається з основних компонентів:

1. Визначення оптимальних шляхів маршрутизації.
2. Передача пакетів інформації через об'єднану мережу.

У даній роботі останній з цих двох компонентів буде називатися комутацією. Комутація є відносно простою, але з іншого боку, визначення маршруту може бути доволі складним і важким процесом.

При визначенні маршруту можна керуватись різними показниками:

- величинах, які є результуючими алгоритмічних обчислень за окремою змінною (наприклад це може бути довжина маршруту);
- сукупністю чи множиною комбінацій певних показників.

Програмні реалізації алгоритмів маршрутизації вираховують показники маршруту для визначення оптимальних маршрутів до кінцевого пункту призначення.

З метою спрощення процесу визначення маршруту, алгоритми маршрутизації можуть бути ініціалізованими і з підтримкою таблиці маршрутизації, в яких міститься маршрутна інформація. При цьому маршрутна інформація може змінюватись в залежності від того який алгоритм маршрутизації буде використовуватись.

3.2 Розробка структурної схеми системи

Розглянувши попередні дані та класифікації можна встановити основні елементи та програмні структури, які потрібні для проектування та розробки програмного забезпечення системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана, яке базується на імітаційній моделі клієнт-сервер, або моделі WEB-сервер – WEB-клієнт.

Ця модель відповідає всім механізмам та алгоритмам які використовуються у сучасних комп’ютерних системах обміну інформації.

На першому етапі побудуємо узагальнюючу структурну схему розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана (рис.3.1):

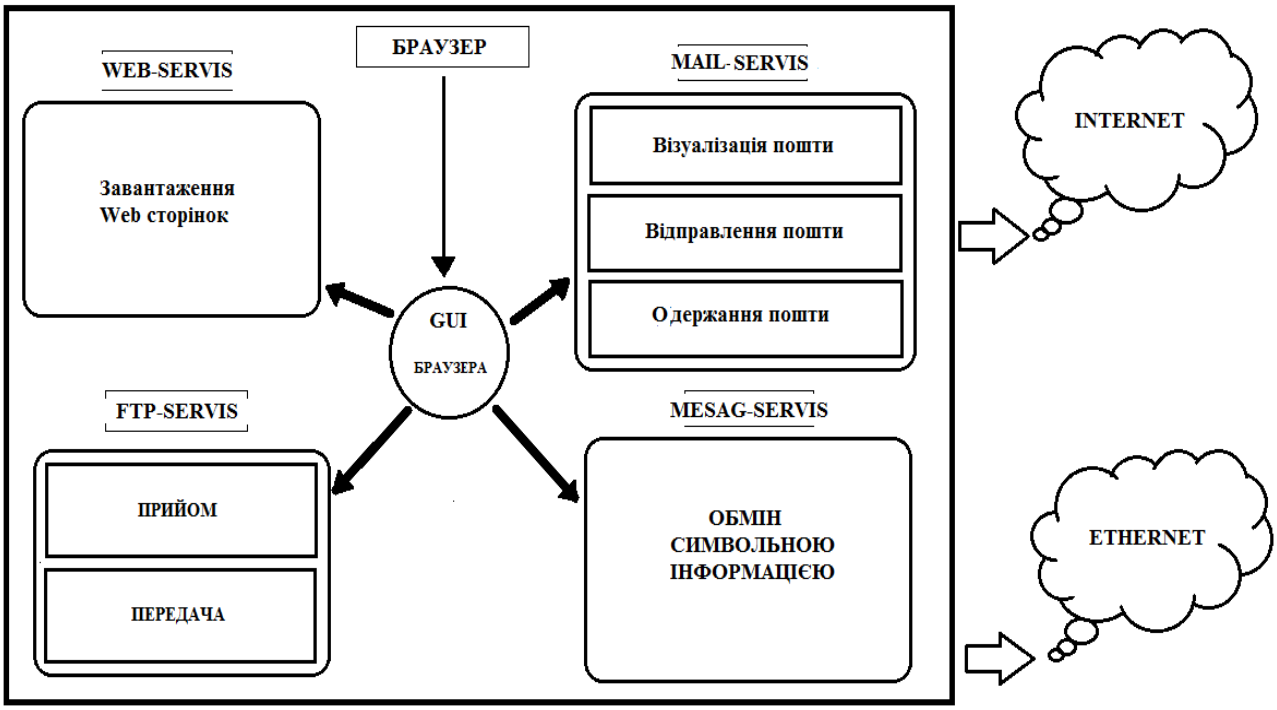


Рисунок 3.1 - Структурна схема системи

Згідно з наведеною схемою можна стверджувати, що програмний засіб, який розробляється повинен складатися з 5 ти основних модулів верхнього рівня (головна програма) і одного низького – взаємодії апаратної частини з мережею (а саме, мережевої взаємодії на рівні сокетів ОС).

Тоді перший рівень такої системи, згідно з базовими основами проектування сучасних мережевих систем, заснованих на GUI повинен бути оснащений окремим модулем, який має бути пов’язаний не з архітектурою самої комп’ютерної системи і навіть не операційною системою, а з узагальненою сучасною уявою звичайного пересічного користувачам про взаємодію структури показаної на рис. 3.2:

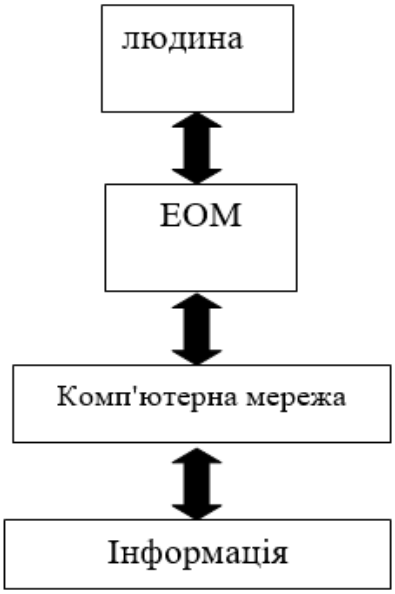


Рисунок 3.2 – Структура доступу користувача до інформації

Цей рівень не має залежати ні від програмної, ні від мовної платформи і повинен дозволяти розробнику-програмісту ефективно і швидко перебудовувати систему з однієї платформи на іншу при зміні вимог до програмної продукції.

Два рівні – WEB та FTP є рівнями стандартного обміну, якими забезпечується зчитування інформація у вигляді (для WEB – гіпертексту) і у файловому (для FTP). Різниця полягає лише в інтерпретації отриманих даних на рівні операційної системи ОС() (або механізмів ОС).

У більшості сучасних ОС перетворення WEB відбувається за більш складним механізмом відтворення сторінок, де кожному окремому елементу надається своя власна система перекодування, але завчасна попередня обробка інформації (приймання і передача) залишається все одно FTP.

Інші наступні рівні за своєю структурою та протоколами обміну інформацією відрізняються не тільки структурою передачі пакетів, але і системою збереження, захистом та реєстрацією даних. Тому вони вимагають окремих стандартних або нестандартних програмних заходів зберігання та обробки інформації.

Алгоритми маршрутизації заповнюють відповідною інформацією маршрутні таблиці. Асоціаціативні команди, такі як "Пункт призначення-наступне пересилання" надають повідомлення роутеру, а певний визначений пункт призначення може бути досягнутий оптимально шляхом відправки пакета в певний вибраний роутер, що являтиме собою наступне пересилання в ланцюзі до кінцевого пункту призначення. Під час прийманні пакетів, роутер проводить перевірку адреси пункту призначення і намагається узгодити отриману адресу з наступним пересиланням.

У таблицях маршрутів може міститися і будь-яка інша інформація. Показники забезпечують інформацію про пріоритетність якогось каналу або тракту. Роутери проводять порівняння показників, з метою оптимізації того чи іншого маршруту. Показники мають відміності один від одного в залежності від використовуваної схеми алгоритму маршрутизації.

Роутери проводять обмін інформацією між собою шляхом передачі різних повідомлень і підтримують свої маршрутні таблиці. Прикладом такого обміну інформацією (чи повідомлення є повідомлення про маршрутні оновлення. Оновлення маршрутизації містять всю таблицю маршрутів або її певну визначену частину. Також проводиться аналіз інформації про відновлення маршрутизації, який надходить від усіх роутерів, і будь-який з них може створити детальну картину топології мережі.

Іншим прикладом повідомлень, з допомогою яких спілкуються роутери, є оголошення про стан каналу, яке інформує інші роутери про стан каналів відправника пакетів. При цьому канальна інформація також може бути використана і для побудови повної картини топології мережі., Роутери можуть визначити оптимальні маршрути до пунктів призначення після того, як топологія мережі стає прозорою і зрозумілою

Для більшості протоколів маршрутизації алгоритми комутації порівняно прості і в основному однакові. А головна обчислювальна машина (ГОМ) визначає необхідність відправки пакета в іншу головну обчислювальну машину у більшості випадків. Отримавши певним способом адресу роутера, ГОМ -

джерело відправляє пакет, який спеціально адресований в фізичну адресу роутера (рівень MAC), однак з адресою протоколу (мережевий рівень) ГОМ пункту призначення.

Адреси протоколу пункту призначення пакета перевіряються роутером. Після цього він визначає спосіб передачі цього пакета наступному роутеру. У випадку. коли роутер не визначився зі способом передачі пакет у більшості випадків ігнорується. У першому випадку роутер передаєє пакет до наступного роутера з допомогою заміни фізичної адреси пункту призначення на фізичну адресу наступного роутера і наступної функції передачі пакета.

Наступне пересилання може відбуватись або не відбуватись. Якщо не відбувається, то наступним пересиланням, як правило, є інший роутер, що бере на себе такий же процес прийняття рішення про комутацію. Орієнтуючись на те, як відбувається просування пакету через об'єднану мережу, його фізична адреса змінюється, але незмінною залишається адреса протоколу.

У викладеному вище описі розглянута комутація між системою кінцевого пункту призначення і джерелом. ISO (Міжнародна Організація зі Стандартизації) провела ряд досліджень, результатом чого стала розробка термінологічної ієрархії, яка стала корисною при описі цього процесу. Якщо користуватися їх термінологією, то мережеве устаткування, яке не володіють спроможністю пересилати пакети при підмережевому обміні, називаються кінцевими системами (ES). А в той час як мережеві пристрої, що володіють такою здатністю, називаються проміжними системами (IS).

В свою чергу проміжні системи поділяються на системи, з можливістю повідомлятися в межах доменів маршрутизації, і системи, які можуть повідомлятися і у межах домену маршрутизації, і з іншими доменами маршрутизації. Часто вважають, що домен маршрутизації є частиною об'єднаної мережі, яка знаходиться під загальним адміністративним управлінням і регулюється певними визначеними наборами керуючих адміністративних принципів. Ще домени маршрутизації називають автономними системами (AS). Для певних визначених протоколів домени

маршрутизації крім того можуть бути додатково розділені на ділянки маршрутизації. Але для комутації всередині ділянок і між ділянками також можна скористатись внутрішньодоменними протоколами визначення маршрутів.

3.3 Розробка функціональної схеми системи

Програмна система, що розробляється, пов’язана з передачею та прийманням інформаційних даних в мережі матиме велику кількість апаратних і програмних зв’язків на рівні апаратної частини та ОС.

Проблеми пов'язі з апаратурою вирішуються за допомогою засобів низького рівня, а саме драйверів і часто пов’язані з використанням API функцій ОС, а вони у свою чергу прив’язані до відповідних їм компіляторів призначених для різних типів ОС.

Побудуємо функціональну схему нашої системи (рис.3.2), яку створено у відповідності зі структурною схемою. Функціональній схемі притаманні всі ті ж елементи, але вона доповнена функціональними зв’язками вирішуючими приналежність та послідовності обробки надходження даних.

Детальне визначення структурних одиниць та їх взаємодія дозволить структурувати програмні залежності, а також визначити програмні модулі, які можна створити виходячи з термінів сучасних технологій програмування, а саме об’єктного програмування.

Функціональну схему системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана показано на рисунку 3.3:

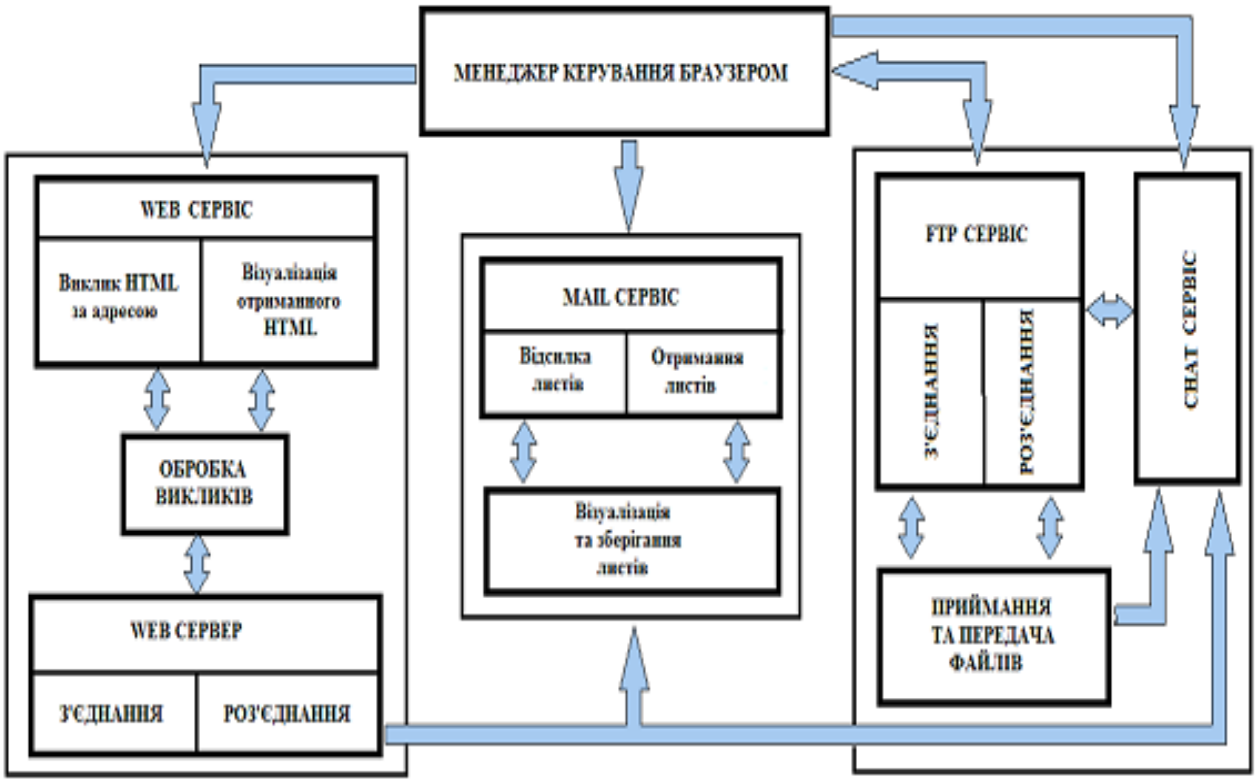


Рисунок 3.3 - Функціональна схема системи

Згідно з рисунком 3.3 більш детальний аналіз структури дозволяє виділити чотири основні класи програми та функції їх взаємодії:

- клас форми;
- клас WEB сервісу;
- клас FTP сервісу;
- клас Mail сервісу.

Причому кожен клас можна підрозділити на окремі підкласи. Клас форми сервісу передбачає створення об’єктів:

- меню клавіш та викликів підзадач для їх викликів;

- меню обробки мережевих викликів, обробки системних помилок та допоміжних засобів допомоги (help, example, та інше);

- графічне GUI супроводження виконання задач браузера.

Клас WEB сервісу передбачає створення об`єктів:

- сокетів клієнт – сервер;

- сбробки імен адрес з’єднання.

- перетворення прийнятих і оброблених даних у відповідному форматі;

- захищення даних при прийнятті;

- з'єднання та від’єднання.

Клас Mail сервісу, який є підкласом WEB (на рівні Socets) передбачає створення об`єктів:

- БД листування;

- редагування (написання) листів;

- обробка прикріпленої інформації.

Клас FTP завдяки своїм особливостям, може і не потребує великих програмних витрат на окреме створення об’єктів, тому. що використовує системні API ОС, але все ж вимагає додаткової уваги з позиції захисту самої системи.

Мережеві адреси визначено комітетом IEEE, який і закріпив за кожною фірмою - виробником таких плат деякі інтервали адрес. Ці адреси безпосередньо «зашиті» всередині корпусів мікросхем. Завдяки цьому кожна плата і кожен комп'ютер мають унікальну мережеву адресу. Плата мережевого адаптера генерує запит у комп'ютера про дані з пам'яті по шині, і якщо вони надходять швидше передаючої спроможності плати, то дані тимчасово переміщуються в буфер.

Перед передачею даних між платою призначеною для передачі інформації і приймаючою платою відбувається діалог з допомогою якого здійснюється узгодження інформації про наступні параметри передачі:

- Максимальний розмір блоку даних для передачі.

- Обсяг даних, що передані без підтвердження про отримання.

- Часові інтервали між передачами блоків даних.

- Часовий інтервал, протягом якого необхідно надіслати підтвердження.

- Обсяг даних, які може прийняти кожна плата без переповнення.

- Швидкість передачі даних.

Такі протоколи - це набір правил і процедур, що призначені для встановлення зав’язків у мережі.

Web було створено для поширення гіпертекстових документів в привабливому графічному форматі, але тепер загальною турботою стало розширення її функціональних можливостей. Користувачі прагнуть персоніфікувати вміст сторінок. Багато компаній хочуть, щоб сторінки Web могли отримувати інформацію з баз даних і складати звіти в задається користувачем формі.

Функції, які виконують Web-сервери:

- прийом запиту від Web-браузера по протоколу HTTP з використанням TCP/IP;
- пошук і відсилання файлу гіпертексту або документа в браузер по HTTP;
- обслуговування запитів mailto, ftp, telnet та ін .;
- запуск додатних програм на Web-сервері з передачею і поверненням параметрів обробки через інтерфейс CGI;
- обслуговування навігаційних карт зображення imagemap;
- служить джерелом завантажуваних програм на мові Java;
- контроль доступу на основі імен і паролів доступу;
- ведення реєстраційного журналу;
- адміністративне та оперативне управління сервером.

Web-браузери спілкуються з Web-серверами через протокол передачі гіпертекстових повідомлень (HypertextTransferProtocol, HTTP), простий протокол запитів та відповідей для пересилання інформації з використанням TCP/IP. Web-сервер отримує запит, знаходить файл, посилає його браузеру і потім розриває з'єднання. Наявна на сторінці графіка обробляється точно так же. Потім настає черга браузера вивести на екран завантажений з мережі HTML-документ.

Хоча зазвичай Web-сервери містять HTML-сторінки і графіку, на них можуть зберігатися будь-які файли.

На Web-сервері виконуються також додатні програми, найбільшого поширення серед яких отримали процесори пошуку та засоби зв'язку з базами даних. Для їх розробки застосовуються такі стандарти, як загальний шлюзовий інтерфейс (CommonGatewayInterface, CGI), мови сценаріїв, подібні JavaScript, а також повноцінні мови програмування, як, наприклад, Java і VisualBasic. Крім CGI кілька постачальників Web-серверів розробили інтерфейси прикладного програмування (API) - як, наприклад, NetscapeServerAPI і InternetServerAPI, створені фірмами Microsoft і ProcessSoftware, - які дозволяють розробникам безпосередньо звертатися до конкретних функцій Web-сервера.

Таким чином, можна зробити висновок, що проектована система відповідає сучасним вимогам до технології створення сучасних програмних засобів.

3.4 Розробка діаграми процесів

Діаграма взаємодії процесів розробленої системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана наведена на рисунку 3.4.

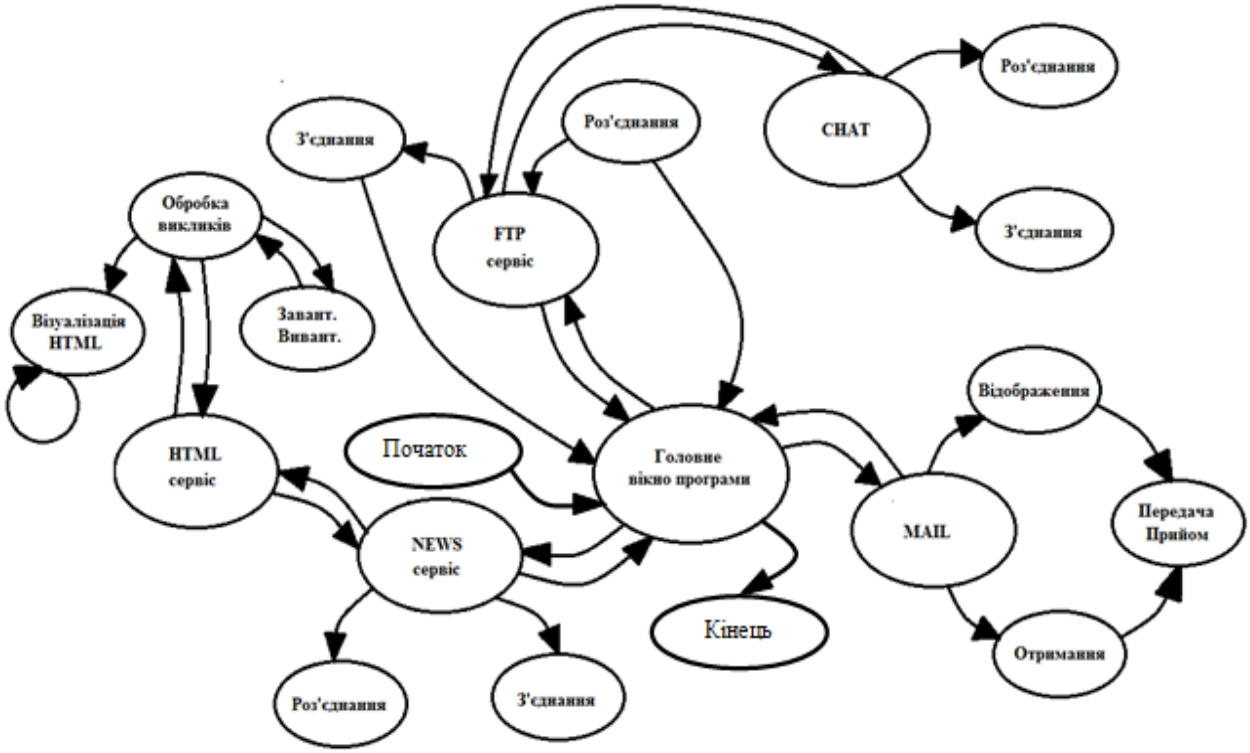


Рисунок 3.4 - Діаграма взаємодії процесів

При детальному її розгляді можна побачити як саме відбувається взаємодія у розробленій системі.

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

Діаграма взаємодії процесів використовується для візуалізації процесів обробки даних (структурне проектування). Для розробника вважається звичним спочатку креслити діаграму взаємодії процесів даних рівня контексту, завдяки чому буде показано взаємодію системи.

Ця діаграма в подальшому підлягає уточненню шляхом деталізації процесів та потоків даних з метою показати систему що розробляється.

З рисунку 3.4 видно, що процеси взаємодіють наступним чином:

- початок;
- завантаження Web-сторінок;
- проведення операцій з відправлення та одержання пошти;
- проведення обміну файлами по протоколу FTP;

- проведення обміну символьної інформацією між комп'ютерами;

- кінець.

У комп'ютерній мережі з архітектурою клієнт-сервер спеціалізований комп'ютер (або виділений сервер) використовується для встановлення всіх розподілених ресурсів. І таке рішення пришвидшує доступ користувачів до централізованих ресурсів мережі і має ряд особливостей:

- адміністрування мереж є значно простішим за рахунок невеликої кількості серверів в мережі і їх вузькоспрямовану спеціалізацію;

- високі вимоги до виділеного сервера - для забезпечення високої продуктивності на сервері потрібна велика кількість оперативної пам'яті, диск великої ємності і процесор з високою продуктивністю;

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

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

4.1 Розробка блок-схем та опис алгоритмів функціонування системи

Під час виконання завдань поставлених у магістерській роботі було створено блок-схеми основної програми та підпрограм. Перед їх розглядом необхідно провести роз’яснення який саме тип блок-схем використовується.

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

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

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

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

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

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

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

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

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

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

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

Блок-схеми є першоджерелами стратегії розвитку ПЗ. Тому від точності і детальної блок-схеми залежить результат всієї програми.

Згідно з розробленою функціональною та структурною схемою розробимо відповідну блок-схему алгоритму роботи основної програми рис.4.1:

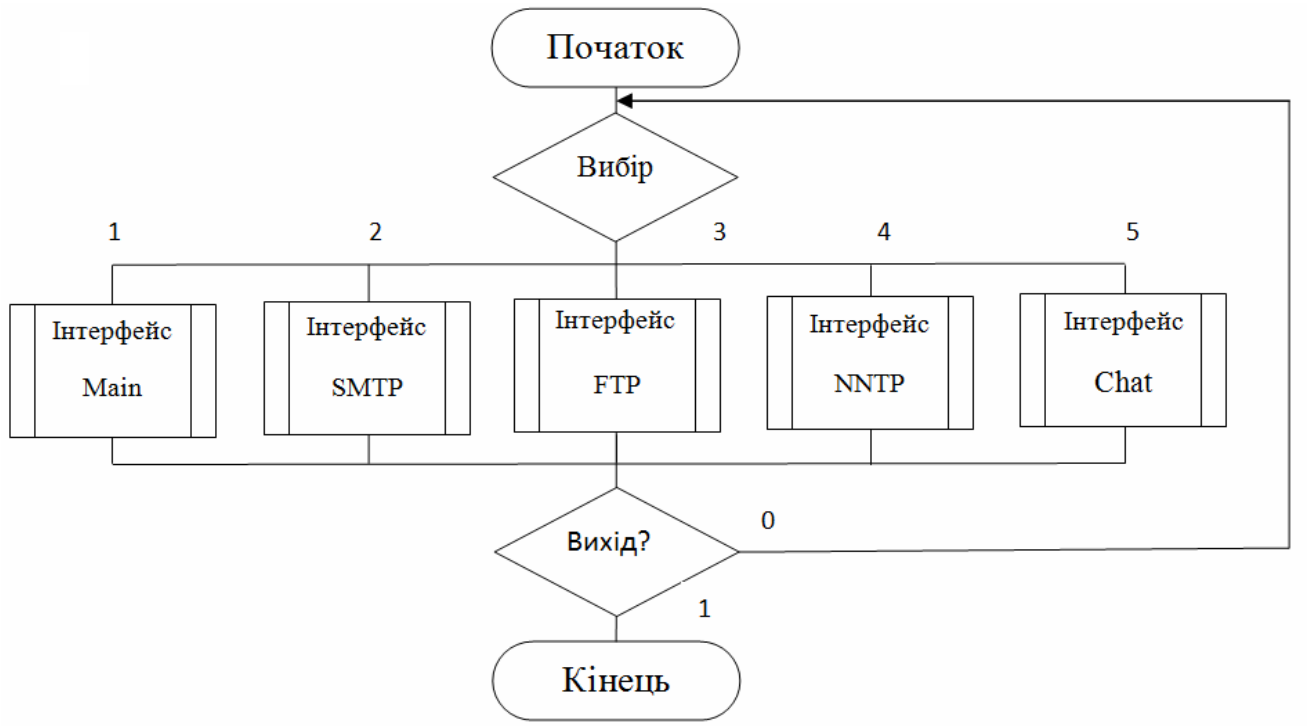


Рисунок 4.1 - Блок-схема основної програми

При розробці WEB-сервера потрібно виділити як мінімум три рівні проектування:

- програмно-апаратний рівень;
- програмний рівень мови та ОС;
- рівень інтерфейсу користувача(GUI).

Програмно-апаратний рівень у більшості сучасних программ високого рівня приховано у класах та бібліотеках компіляторіві і представляють цікавість тільки з точки зору низькорівневого програмування на рівні ОС при доступі до функцій конкретної ОС.

Блок-схему підпрограми представлено на рисунку 4.2:

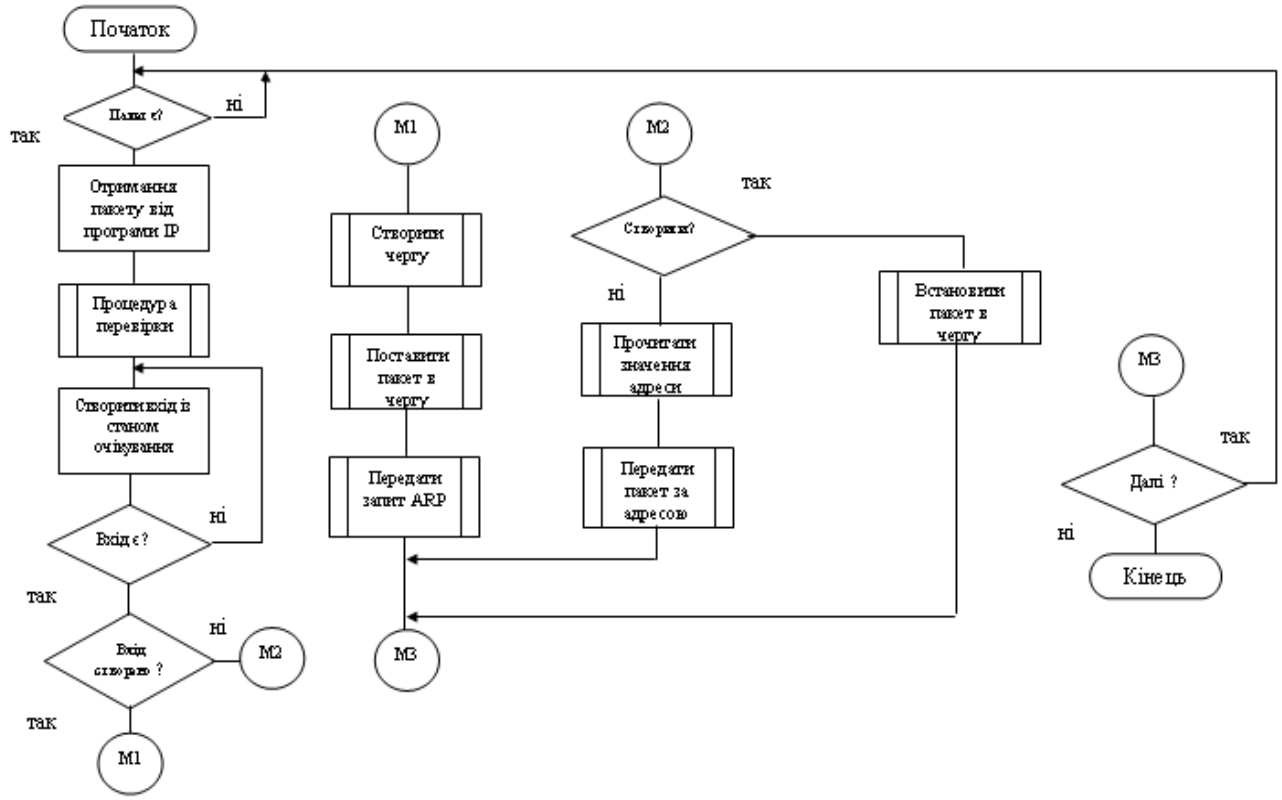


Рисунок 4.2 - Блок-схема підпрограми

Програмний рівень мови програмування реалізовано на базі компонентів программного інтерфейсу і має безперервний зв'язок з методами та классами GUI. Таким чином для створення повноцінного WEB-достатньо механізмів і компонентів мови програмування, у нашому випадку DELPHI(LAZARUS).

Будемо створювати наш браузер на базі компонента - WebBrowser (Internet).

Встановлюємо на форму: один компонент WebBrowser зі вкладки Internet, компоненти SpeedButton, компонент ComboBox, компонент Panel, компонент ProgressBar, компонент StatusBar, компонент OpenDialog і компонент SaveDialog; в uses підключимо MSHTML і ActiveX.

Створимо подію OnCreate на формі і запишемо в ньому наступний код:

```
begin
    WebBrowser1.Navigate('http://www.google.com');
end;
```

Пропишемо кнопку "вперед". Це кнопка, яка будуть натискати після введення адреси в ComboBox.

Для цього:

- створимо подія OnClick на SpeedButton6 наприклад, і в коді напишемо:

```
var
a: string;
begin
WebBrowser1.Navigate (ComboBox1.Text); // Передаємо в пошук, зазначену адресу з ComboBox1;
a: = ComboBox1.text; // Записуємо паралельно цю адресу в змінну;
ComboBox1.Items.Add (a); // Завантажуємо введене в список ComboBox1;
end;
```

Наведений код, дозволятиме нам при натисканні на ComboBox, переглядати історію введених адрес які вводяться вручну, в рядок браузера.

Для зручності зробимо так, щоб механізм працював не тільки від SpeedButton6 але і від натискання на клавішу Enter, для цього:

- створюємо подія OnKeyPress, у компонента ComboBox1 запишемо наступне:

```
var
    a:string;
begin
    if Key=#13 then
    begin
      WebBrowser1.Navigate(ComboBox1.Text);
       a:=ComboBox1.text;
      ComboBox1.Items.Add(a);
    end;
end;
```

Тепер виконувати пошук в інтернеті можна і після натиснення клавіші Enter.

Створимо клавіші - крок назад і крок вперед. Для клавіші - Крок назад, пишемо в події OnClick у кнопки SpeedButton1:

```
var
      Doc, Histor: Variant;
begin
     Doc:= WebBrowser1.OleObject.document;    //Посилаємося на інтерфейс документу;
          if TVarData(Doc).VDispatch = nil then Exit;     //Якщо документ відсутній, тоді виходимо;
          Histor:=Doc.parentWindow.history;      {Об’єкт документа – об’єкт батьківського вікна – об’єкт містить історію переходів. Крок назад до попереднього URL.};
          Histor.go(-1);
```

```
end;
```

Для клавіші - Крок вперед пишемо в події OnClick у кнопки SpeedButton2:

```
var
      Doc, Histor: Variant;
begin
      Doc:= WebBrowser1.OleObject.document;   //Посилаємося на інтерфейс документа;
       if TVarData(Doc).VDispatch = nil then Exit;    //Якщо документ відсутній, тоді виходимо;
      Histor:=Doc.parentWindow.history;      {Об’єкт документа – об’єкт батьківського вікна – об’єкт який містить історію переходів крок вперед до попереднього URL.};
      Histor.go(1);
end;

Для зупинки завантаження в OnClick на SpeedButton 4 пишемо;

begin
    WebBrowser1.Stop;
end;
```

Для оновлення сторінки в OnClick на SpeedButton 3 пишемо;

```
begin
WebBrowser1.Refresh;
end;
```

Переходимо до наступного: пропишемо ряд клавіш, що надають нашому додатку додаткову функціональність.

Наприклад, забезпечимо відкриття файлів в нашому браузері розташованих на локальних дисках.

Для цього в OnClick на SpeedButton7 пишемо:

```
begin
     if OpenDialog1.Execute then
    begin
       WebBrowser1.Navigate(OpenDialog1.FileName);
       ComboBox1.Text:=OpenDialog1.FileName;
   end;
end;
```

Для можливості браузера зберігати сторінки на комп'ютері, в OnClick на SpeedButton9

```
WebBrowser1.ExecWB(OLECMDID_SAVEAS,
OLECMDEXECOPT_ DODEFAULT);  //Зберігаємо вміст WebBrowser1;
```

Для отримання можливості друку після натискання на SpeedButton 8 в OnClick останнього пропишемо:

```
var
     PostData, Headers: OLEvariant;
begin
    WebBrowser1.ExecWB(OLECMDID_PRINT, OLECMDEXECOPT_DODEFAULT,  PostData,
Headers);  //Друкуємо вміст  TWebBrowser;
end;
```

Для попереднього перегляду сторінки перед друком, в OnClick на SpeedButton 10 пишемо:

```
begin
     WebBrowser1.ExecWB(OLECMDID_PRINTPREVIEW,   OLECMDEXECOPT_DODEFAULT);
//Попередній перегляд;
end;
```

Розробка блоку FTP-сервера

Для створення FTP сервера скористаємося компонентом ICS. Для реалізації цього завдання використовуємо компонент FtpServer.

У цього компонента можна змінювати такі властивості:

- Bunner - це текст, який побачить користувач, коли підключиться до сервера по FTP-протоколу;

- Maxciients - максимальна кількість клієнтів, які одночасно підключені до сервера;

- Port - порт, на якому буде працювати сервер. За замовчуванням зазвичай ftp дорівнює 21-му порту. Але можна змінити це значення, якщо даний порт вже зайнятий.

Для запуску сервера FTP досить тільки викликати метод start компонента FtpServer.

Приклад коду, який виконується при натисканні кнопки Запустити:

```
procedure TMainForm.StartButtonClick(Sender: TObject);
begin
     FtpServerl.Banner := BannerEdit.Text;
     FtpServerl.MaxClients := SpinEditl.Value;
    FtpServerl.Port := IntToStr(SpinEdit2.Value);
    FtpServerl.Start;
end;
```

У цьому коді встановлюються описані властивості сервера, і він запускається. Для зупинки сервера виконується наступний код:

```
procedure TMainForm.StopFTPlClick(Sender: TObject);
begin
    if bConnected = true then
   begin
      FtpServerl.DisconnectAll;
      FtpServerl.Stop;
   end;
end;
```

Тут відбувається перевірка: якщо сервер запущений, то спочатку потрібно відключити всіх користувачів за допомогою методу DisconnectAll, а потім зупинити сервер методом stop. Бажано завжди відключати користувачів від сервера перед зупинкою, щоб не виникало помилок.

Будь-які серверні додатки повинні бути надійні і перевіряти будь-які нестандартні ситуації, щоб не викликати збої в роботі самого сервера і ОС в цілому. Для цього програміст повинен діяти, виходячи з мінімальних припущень. Будь-який параметр, що передається програмі, повинен перевірятися на коректність. Якщо клієнт передасть невірний параметр, програма обробить його некоректно, тоді для сервера все може закінчитися збоєм. Як мінімум програма зависне, а як максимум можуть зіпсуватися дані на жорсткому диску сервера.

У компонента FtpServer потрібно обробляти досить багато подій. Одна з них - Authenticate. При її спрацьовуванні включається наступний набір операторів:

```
procedure TMainForra.FtpServerlAuthenticate(Sender: TObject;
Client: TFtpCtrlSocket; OserName, Password: TFtpString; var Authenticated:
Boolean);
begin
      if isClientThere(UserName) = false then
     begin
          clidir := isClient(username,password,client);
         if clidir <> '' then
        begin
           Authenticated := true;
          client.HomeDir := clidir;
         end;
    end
    else
```

```
            Authenticated := false;
           StatusBarl.Panels[1].text   :=   'Кількість   користувачів:   '   +
IntToStr(ListViewl.Items.count);
      end;
```

Ця подія спрацьовує кожного разу, коли користувач підключається до порту сервера, і потрібно перевірити коректність введених імені користувача і пароля. Як параметри в цей обробник ми отримуємо наступні значення:

- UserName - ім'я користувача, з яким підключається клієнт;

- Password - пароль, який вказав користувач;

- Authenticated - якщо після виходу з процедури ми встановимо цю властивість рівною true, то користувач зможе працювати з сервером. Якщо встановити false, це буде означати, що користувач не пройшов аутентифікацію і ми відхилили з'єднання.

Далі використовується функція isciient. Функція повертає домашню директорію користувача, що підключається. Якщо вона поверне порожній рядок, то ми повинні відхилити з'єднання, тому що не можна працювати з FTP-сервером без вказівки директорії. Можна додати директорії за замовчуванням, яка буде використовуватися в тих випадках, коли не вказано іншого значення.

Наступна подія, яку потрібно обробити, - OnCh'angeDlrectory. Ця подія генерується, коли користувач намагається перейти в іншу директорію. Код обробника події:

```
procedure TMainForm.FtpServerlChangeDirectory(Sender: TObject;
Client: TFtpCtrlSocket; Directory: TFtpString; var Allowed: Boolean);
begin
    if length(Client.Directory) < length(client.HomeDir) then
    Allowed := false
else
   Allowed := true;
   if pos('..', Client.Directory)>0 then
   Allowed := false;
end;
```

Тут відбувається проста перевірка по довжині шляху. Якщо поточний шлях менше директорії, встановленої за замовчуванням для клієнта, то змінній Allow присвоюється значення false, тому що не можна підніматися вище рівня домашньої директорії. Наприклад, якщо домашня директорія з: \Home, то шлях C:\ виявляється менше і на нього переходити не можна.

Ця перевірка дуже проста і вона не захищена. Користувач може написати такий шлях: C:\Home\ .. \, що дозволить йому піднятися вище при тому, що формально шлях довший. Саме тому крім вимірювання довжини ви повинні перевіряти наявність в шляху точок.

Якщо точки присутні, то шлях невірний, і потрібно заборонити перехід за цією адресою. У нормальному шляху можуть бути тільки одинарні точки, подвійні заборонені.

Наступна подія, яке обробляється, - Onciientcommand. Ця подія генерується кожен раз, коли користувач намагається виконати якусь FTP-команду. Тут потрібно перевіряти, чи має права користувач завантажувати, вивантажувати, перейменовувати файли.

Нижче наведено приклад коду, який перевіряє дозволи на виконання різних команд:

```
       procedure TMainForm.FtpServerlClientCommand(Sender: TObject;
       Client: TFtpCtrlSocket; var Keyword, Params, Answer: TFtpString);
       var
           SFD1 : String;
           SFD2 : String;
       begin
            ModifyClient(client.username,Keyword,client.directory);
            Logit(client.UserName + ' - ' - ) - client.DataSocket.Addr + ' ' +
Keyword + ' ' +
           client.directory + params);
           if (Keyword = 'PUT') or (Keyword = 'STOR') then
           begin
              if IsAllowedTo(client.username,2) = false then
             begin
               client.SendAnswer('501 - не дозволено!');
               exit;
             end;
           end;
       if (Keyword = 'GET') or (Keyword = 'RETR') then
            begin
               if IsAllowedTo(client.username,3) = false then
              begin
                 client.SendAnswer('501 -  не дозволено!');
                 exit; t
              end;
            end;
       if Keyword ='RNFR' then
            begin
              if IsAllowedTo(client.username,4) = false then
               begin
                  client.SendAnswer('501 - не не дозволено!');
                  exit ;
              end;
           sfdl := client.directory + params;
       end;
       if Keyword = 'RNTO' then
           begin
```

```
        if IsAllowedTo(client.username,4) = false then
           begin
              client.SendAnswer('501 -  не дозволено!');
              exit;
           end;
      sfd2 := client.directory + params;
      FileORDirRNTO(sfdl,sfd2) ;
       sfdl := '';
       sfd2 := '';
   end;
if Keyword = 'DELE' then
begin
      if IsAllowedTo(client.username,5) = false then
         begin
           client.SendAnswer('501 - не дозволено!');
           expend;
           fileordirdel(client.Directory,params);
           client.FileName :='';
           client.Directory := '';
        end;
end;
```

Через параметр Keyword ми отримуємо текстову назву команди, яку хоче виконати користувач.

Це основні події, потрібні для забезпечення безпеки сервера.

Розробка блоку TELNET-клієнта

Розглянемо створення Telnet клієнта. За допомогою такого клієнта можна підключиться до якого-небудь серверного порту і виконувати команди прямо на сервері. Наприклад, якщо підключитися до порту Telnet, то можна запускати на сервері програми.

Якщо підключиться до порту FTP, то можна виконувати команди FTP з командного рядка.

Для тестування нашого Telnet-клієнта будемо використовувати підключення до локального FTP-сервера, який входить в поставку WindowsXX.

Створимо два рядки введення:

- для введення IP-адреси сервера, до якого ми хочемо підключитися;

- для введення номера порту, який треба використовувати.

В єдиному списку, що випадає в TComboBox можна вибирати тип терміналу, що використовується.

Крім цього нам знадобляться дві кнопки: Connect і Disconnect для підключення і відключення від сервера. І один компонент TMemo, в якому ми

будемо набирати команди, що відправляються серверу, і відображати результат виконання команд.

Самим основним компонентом буде idTelnetDemo з закладки Indy Clients палітри компонентів.

У обробнику натиснення кнопки Connect пишемо наступний код:

```
procedure TTelnetForm.ConnectButtonClick (Sender: TObject);
begin
      IdTelnetl.Terminal:=TerminalCB.Text;
      IDTelnetl.Host := EdtServer.Text;
      IDTelnetl.port := SpnedtPort.Value;
     IdTelnetl.Connect;
end;
```

У перших трьох рядках встановлюються властивості компонента IdTelnetl:

- Terminal - тут вказується у вигляді тексту тип термінал, що використовується;

- Host - адреса сервера, до якого будемо підключатися;

- Port - номер порту, до якого потрібно робити підключення.

Оброблювач натиснення кнопки Disconnect:

```
procedure TTelnetForm.btnDisconnectClickfSender: TObject);
begin
       IdTelnetl.Disconnect;
end;
```

Тут викликається метод Disconnect компонента IdTelnetl, що призводить до відмови у доступі до сервера.

Для відправки команд на сервер ми створимо обробник події onKeyPress для компонента Memo. У цьому обробнику напишемо наступне:

```
procedure TTelnetForm.MemolKeyPress(Sender: TObject;
var Key: Char);
begin
      if IdTelnetl.Connected then
      begin
           IdTelnetl.SendCh(Key);
      end;
end;
```

Тут відбувається перевірка: якщо компонент IdTelnetl підключений до сервера, то символ натиснутої клавіші потрібно передати на сервер. Для цього використовується метод SendCh компонента IdTelnetl, а в якості параметра цього методу передається натиснутий символ. Тепер, при натисканні будь-якої клавіші, щоб ввести символ в компонент Memo, цей символ відправляється на сервер.

Далі створимо обробник події OnConnected для компонента idTelneti. У цьому обробнику напишемо наступний код:

```
procedure TTelnetForm.IdTelnetlConnectedfSender: TObject);
begin
Memol.Lines.Add('Клієнт підключений.');
Memol.Lines.Add('Можете виконувати команди');
Memol.Lines.Add('');
end;
```

Вся робота цього обробника спрямована на те, щоб проінформувати користувача про те, що з'єднання відбулося успішно.

Відповідний текст виводиться в компонент Memo. Тепер створимо обробник події OnDataAvaiiabie компонента idTelneti. Цей оброблювач викликається кожен раз, коли до нас з сервера надійшли дані.

```
procedure TTelnetForm.IdTelnetlDataAvailable(Sender: TIdTelnet;
const Buffer: String);
const
CR = #13;
LF = #10;
var
      Start, Stop: Integer;
begin
     Memol.Lines.Add('');
     Start := 1;
     Stop := Pos(CR, Buffer);
    if Stop = 0 then
        Stop := Length(Buffer) + 1;
while Start <= Length(Buffer) do
      begin
          Memol.Lines.Strings[Memol.Lines.Count - 1] :=
         Memol.Lines.Strings[Memol.Lines.Count - 1] +
        Copy(Buffer, Start, Stop - Start);
         if Buffer[Stop] = CR then
         begin
              Memol.Lines.Add('') ;
         end;
        Start := Stop + 1;
        if Start > Length(Buffer) then
       Break;
        if Buffer[Start] = LF then
       Start := Start + 1;
        Stop := Start;
```

```
          while (Buffer[Stop] <> CR) and (Stop <= Length(Buffer)) do
        Stop := Stop + 1;
      end;
end;
```

Весь написаний вище код спрямований на виведення тексту, який надійшов, за допомогою компонента Memo. Для цього шукаємо символи кінця рядка і перекладу каретки, і якщо вони знайдені, то в компонент додається новий рядок.

Для полегшення пошуку заведені дві константи CR і LF з шістнадцятковим значеннями #13 і #10, які є кодами символів кінця рядка і перекладу каретки.

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ У ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

Розроблене програмне забезпечення системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана має наступні можливості:

1. Завантаження Web-сторінок;

2. Операції з відправлення та одержання пошти;

3. Обмін файлами по протоколу FTP;

4. Можливість обміну символьної інформацією між двома вилученими комп'ютерами, на яких повинен бути запущений даний браузер.

WWW-сервіс:

procedure TMainForm.Exit 1Click - здійснює вихід з програми;

procedure TMainForm.FindAddress - запрошувати HTML-файл за вказаною адресою;

procedure TMainForm.DocumentSource1Click - виводить вихідний текст отриманого HTML-файлу;

procedure TMainForm.StopBtnClick - зупиняє завантаження HTML-файлу;

procedure TMainForm.RefreshBtnClick - перевантажує останній отриманий HTML-файл;

procedure TMainForm.BackBtnClick - викликає попередній завантажений HTML-файл;

procedure TMainForm.ForwardBtnClick - викликає подальший завантажений HTML-файл;

procedure TMainForm.ToolButton2Click - завантажує об'єкт TMail, відповідальний за відправку та отримання пошти;

procedure TMainForm.ToolButton3Click - завантажує об'єкт TMyFtp, відповідальний за обмін даних в протоколі FTP;

procedure TMainForm.ToolButton4Click - завантажує об'єкт TNewsForm, відповідальний за отримання і відправку новин;

procedure TMainForm.ToolButton9Click - завантажує об'єкт TChatForm, відповідальний за обмін символьною інформацією між двома віддаленими комп'ютерами;

FTP-сервіс:

procedure TMyFtp.ConnectBtnClick - з'єднується із зазначеним FTP-сервером;

procedure TMyFtp.Disconnect - обриває з'єднання з FTP-сервером; procedure TMyFtp.CopyItemClick - виконує копіювання вибраного файлу з FTP-сервера; procedure TMyFtp.PasteFromItemClick - відсилає файл на FTP-сервер;

Chat-сервіс:

procedure TChatForm.FileConnectItemClick - виконує запит адреси комп'ютера, з яким буде відбуватися обмін символьною інформацією;

procedure TChatForm.Memo1KeyDown - зчитування символу з клавіатури і відправлення його в сокет клієнта;

procedure TChatForm.Disconnect1Click - розриває з'єднання з віддаленим комп'ютером;

procedure TChatForm.ClientSocketRead - зчитування інформації з віддаленого комп'ютера;

News-сервіс:

procedure TNewsForm.FileConnectItemClick - виконує з'єднання з сервером новин;

procedure TNewsForm.FileDisconnectItemClick - розриває з'єднання з сервером новин;

procedure TNewsForm.NNTP1DocOutput - висновок груп новин;

Mail-сервіс:

procedure TMail.CreateHeaders - створює заголовок для поштового повідомлення;

procedure TMail.SendMessage - відправляє повідомлення;

procedure TMail.SendFile - відправляє файл, "прив'язаний" до листа;

procedure TMail.SMTP1DocInput - введення тексту поштового повідомлення;

procedure TMail.SMTPConnectBtnClick - виконує з'єднання з почтовим сервером.

Для наочності порівняння швидкості роботи запропонованого алгоритму Форда-Белмана при збільшенні розміру мережі, побудуємо графік часу збіжності класичного та вдосконаленого алгоритмів Форда-Белмана . Як видно з графіків, при збільшенні кількості вузлів мережі (X) від 10 до 100, класичний алгоритм Форда-Белмана демонструє стрімке зростання часу обчислення (до 25.4 мс), тоді як розроблений вдосконалений алгоритм завдяки оптимізації утримує стабільні показники в межах 4.1 мс.

Графік часу збіжності класичного та вдосконаленого алгоритмів Форда-Белмана представлено на рис. 5.1:

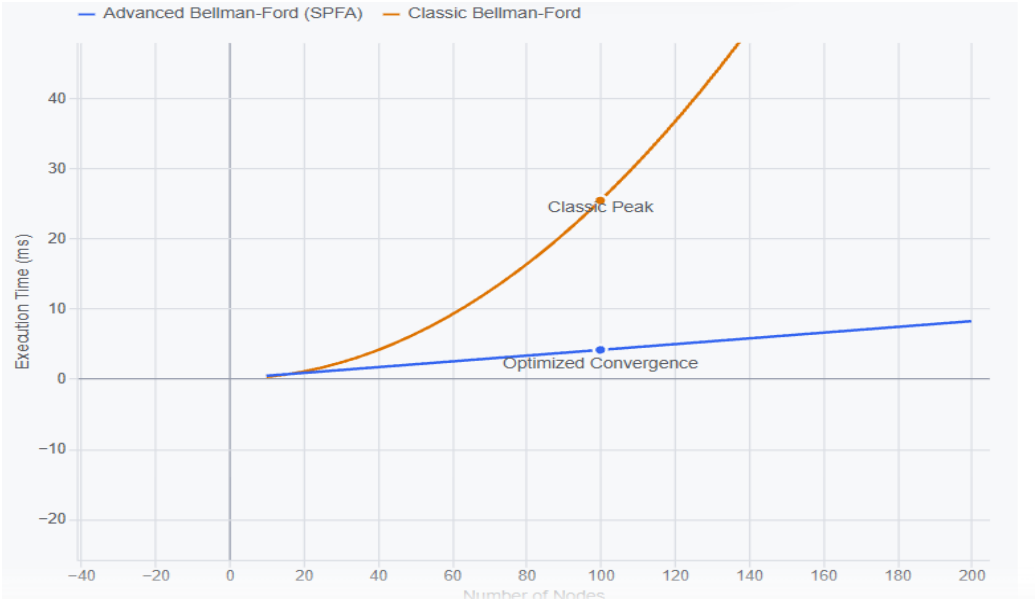


Рисунок 5.1 - Графік часу збіжності класичного та вдосконаленого алгоритмів Форда-Белмана

Розроблена імітаційна модель комп’ютерної мережі для 5-ти ЕОМ показана на рис.5.2:

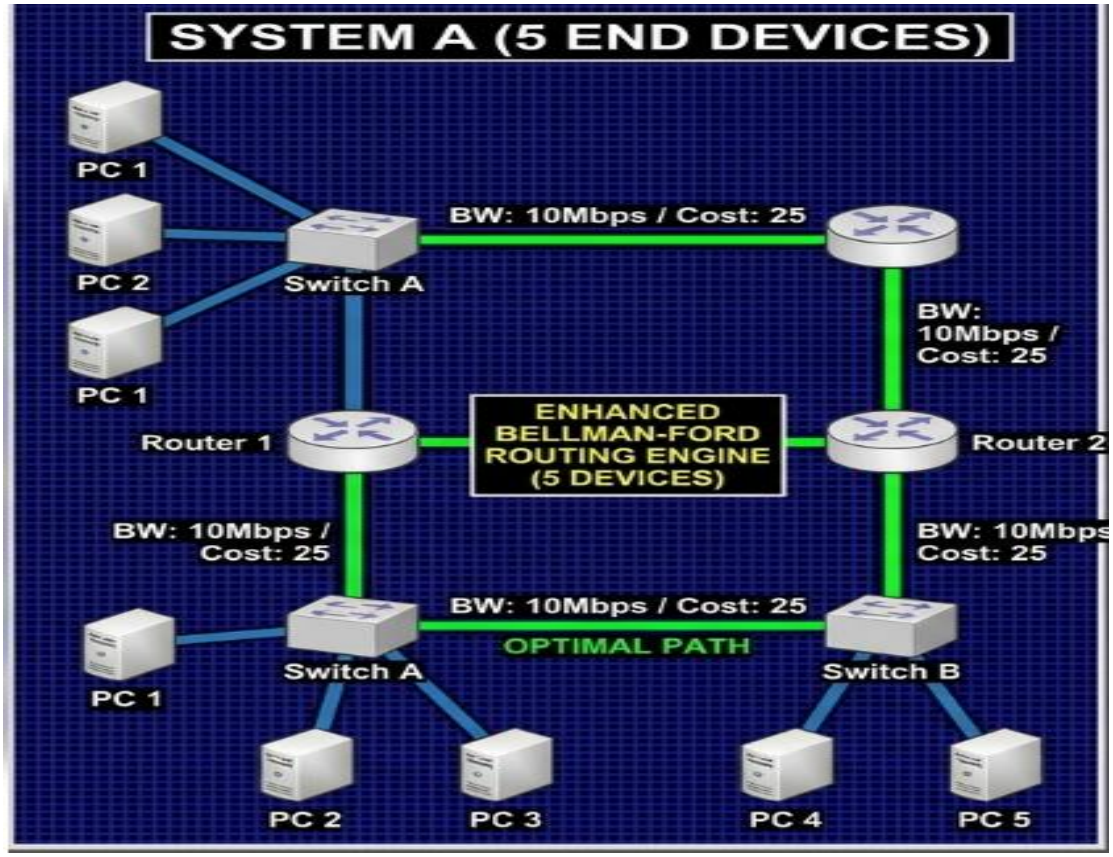


Рисунок 5.2 - Імітаційна модель комп’ютерної мережі для 5-ти ЕОМ

Розроблена мітаційна модель комп’ютерної мережі для 10-ти ЕОМ показана на рис.5.3:

Імітаційна модель комп’ютерної мережі для великої кількості ЕОМ імітує пересилку пакету від одного комп'ютера до іншого.

Потім, при натисканні клавіші ENTER, програма переходить в текстовий режим з UNIX-подібним інтерфейсом, запитуючи користувача адресу одержувача, адреса відправника, і дані типу "String".

Потім кожен комп'ютер або маршрутизатор, по якому проходить пакет, виводить на екран повідомлення про прийом та подальшу відправку пакета адресату і час, в який він отримав і відправив пакет. Оптимальний маршрут розраховується на основі вдосконаленого алгоритму Форда-Беллмана. Програма написана на мові Delphi.

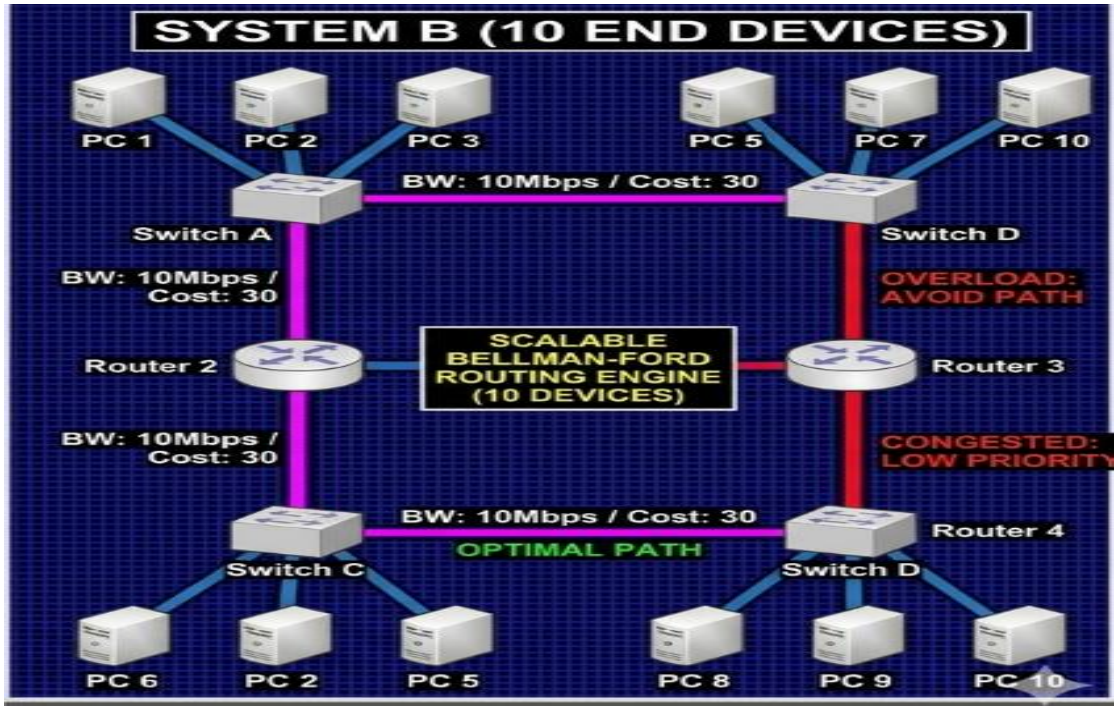


Рисунок 5.3 - Імітаційна модель комп’ютерної мережі для 10-ти ЕОМ

Під час запуску основної програми на екрані виникає схема мережі, показана на рисунку 5.4.

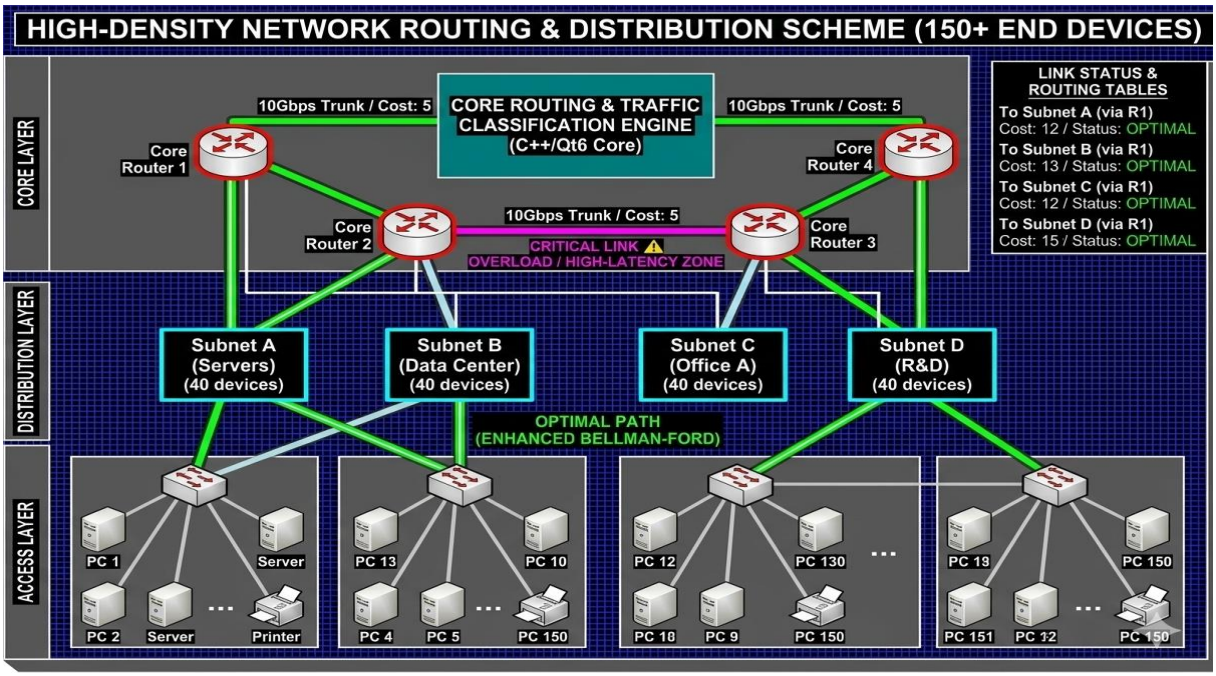


Рисунок 5.4 – Схема комп’ютерної мережі.

На рисунку 5.5 зображено авторські дані розробленого програмного забезпечення:

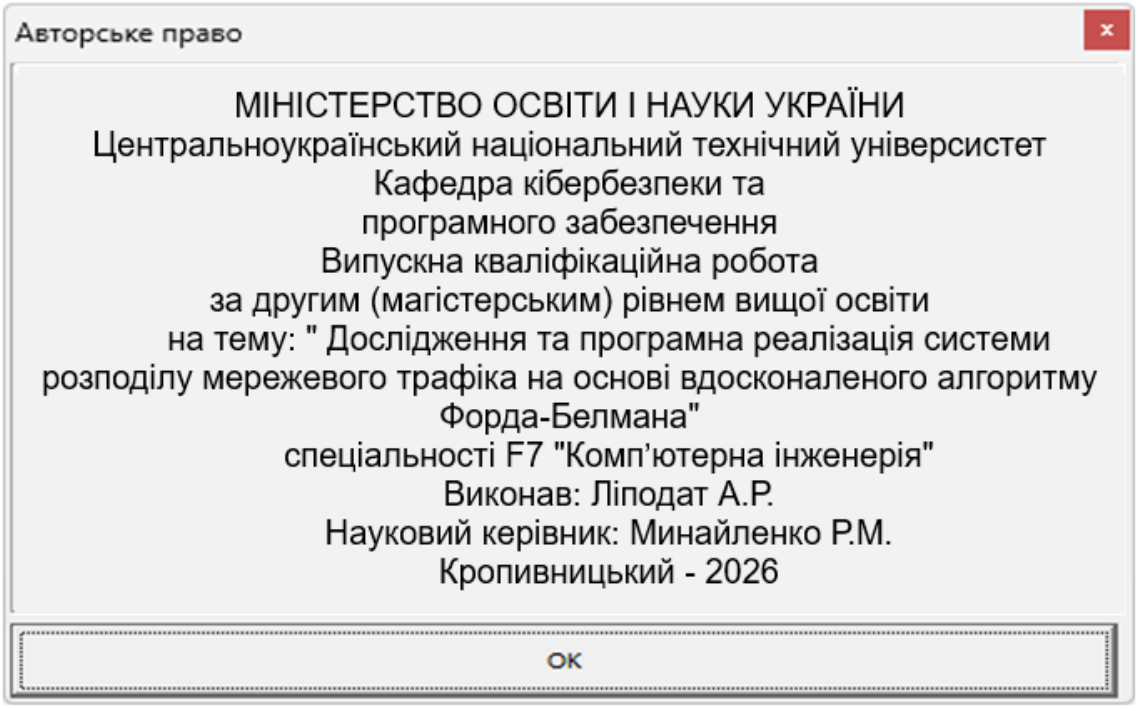


Рисунок 5.5 - Авторське право

Типи даних і змінні основної підпрограми:

```
const AdjacencyMatrix: array [1..VertexQuantity, 1..VertexQuantity] of byte = (
                (0,1,0,1,0,0,0),
                (1,1,1,0,1,0,1),
                (0,1,0,1,0,0,0),
                (1,0,1,0,1,0,0),
                (0,1,0,0,1,1,0),
                (0,0,0,0,1,0,1),
                (0,1,0,0,0,1,0)) - матриця суміжності маршрутизаторів;
TYPE TAddr = record
 router: byte;
 domain: byte;
 comp: byte;
```

END - адреса комп'ютера, що складається з номера маршрутизатора, номера області даного маршрутизатора і номера комп'ютера в цій галузі;

TYPE TBatch = record

from: TAddr;

to_: TAddr;

data: string;

path: array [1..20] of byte; {path is chain of router numbers}

END - пакет, що складається з адреси відправника, адреси одержувача, даних і шляху проходження пакета;

TYPE TComp = object - модель комп'ютера, що складається з адреси, комірки пам'яті для одержання або пересилання пакета;

addr: TAddr;

mem: TBatch;

Procedure Send2Router (batch: TBatch) - процедура посилки пакету на маршрутизатор;

Procedure Send (batch: TBatch) - процедура посилки пакету усередині своєї мережі;

Procedure Receive (batch: TBatch; byRouter: boolean) - прийом пакета;

END;

TYPE TRouter = object - модель маршрутизатора, що складається з номера маршрутизатора, його координат, і комірки пам'яті;

num: byte;

x, y: integer;

memory: Tbatch;

state: boolean;

VAR computers: array [1..38] of TComp - масив комп'ютерів глобальної мережі;

routers: array [1..7] of TRouter - масив маршрутизаторів;

OptimalPath: array [1..49] of byte - оптимальний шлях, розрахований маршрутизатором;

Procedure Receive (routerNum: byte; batch: TBatch) - прийом пакета;

Procedure Send2Comp (batch: TBatch) - відправка пакету у своїй мережі;

Procedure CalcMinPath (sender, target: byte) - обчислення оптимального шляху відправки;

Procedure Send2NextRouter (batch: TBatch; currentRouter: byte) - відправлення на наступний маршрутизатор;

END;

6 НАУКОВА НОВИЗНА

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначене для системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана.

Метою розробки є дослідження та програмна реалізація системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана.

Об'єктом дослідження є системи розподілу мережевого трафіка.

Предметом дослідження є методи створення програмного забезпечення системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана.

Методи дослідження базуються на методах теорії побудови комп'ютерних мереж, методах математичної статистики, методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених метою дослідження, отримані наступні результати:

– Удосконалено систему розподілу мережевого трафіка.

– Розроблено вітчизняний продукт - систему розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана, який має більш широкі можливості, на відміну від існуючих аналогів.

7. ЗАХОДИ З ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

7.1. Вступ

Комп'ютерні мережі вриваються у життя людей в їх професійну діяльність найнесподіванішим і масовим чином. Вони породили істотно нові технології обробки інформації – мережні технології. У найпростішому разі мережні технології дозволяють спільно використовувати ресурси – нагромаджувачі великий ємності, друкують устрою, доступ в Internet, бази й банки даних. Найбільш сучасні і перспективні підходи до мереж пов'язані з допомогою колективного поділу праці – розробці різних документів і майже проектів, управлінні установою чи підприємством і т. д.

Впровадження комп'ютерних технологій принципово змінило характер праці різних категорій фахівців. Працівники, використовують комп'ютерну техніку, на своєму досвіді оцінили її величезні можливості. Одночасно виникла певна безтурботність при її експлуатації.

Недотримання вимог безпеки призводить до того, що й через кілька днів роботи за комп'ютером співробітник починає відчувати певний дискомфорт: в нього виникає головний біль і різь у власних очах, з'являються почуття виснаження й дратівливість. В окремих людей порушується сон, погіршується зір, занедужують руки, шия, поперек тощо.

До недоліків умов праці користувачів комп’ютерної техніки можна віднести:

- недостатню площу і обсяг виробничого приміщення;

- недотримання вимог, мікроклімату на робочих місцях;

- низький рівень освітленості у приміщеннях і на робочих поверхнях апаратури;

- підвищений рівень низькочастотних магнітних полів від моніторів;

- порушення вимог організації робочих місць;

- недотримання вимог до режимам праці та відпочинку;

- надмірне виробничу навантаження працівників;

- відсутність навичок зниження впливу психоемоційного напруги.

Відповідно до ст.14 Закону «Про охорони праці» на роботодавця покладено обов'язок забезпечити: безпеку працівників при експлуатації устаткування; застосування коштів індивідуальної захисту працівників; відповідні вимоги охорони праці, умови праці в кожному робоче місце; дотримання режиму праці та відпочинку працівників; навчання безпечним методам і прийомам виконання; інструктаж з охорони праці; організацію контролю над станом умов праці в робочих місць; проведення атестації робочих місць в умовах праці.

Максимально зменшити кількість шкідливих впливів на людину при високій продуктивності праці, створити комфортні умови для роботи людей – ось одна з головних задач охорони праці.

7.2. Аналіз умов праці

Приміщення розташовано на третьому поверсі п'ятиповерхового будинку. У приміщенні розташовано 3 робочих місця з комп'ютерами (далі ПК). Відповідно до норм «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» ДСанПіН 3.3.2-007-98 площа, що відводиться для робочого місця з комп'ютером повинна бути не менше 6 $м^2$, об'єм не менше 20 $м^3$. Розміри даного приміщень складають: довжина – 6 м, ширина – 4,5 м, висота – 3,5 м, тобто загальна фактична площа складає 27 $м^2$. Необхідна площа на 3 робочих місця із установленими ПК складає 18 $м^2$, що не перевищує фактичну. Обсяг приміщення на одного працюючого складає 31,5 $м^3$, отже відповідає нормі ДСанПіН 3.3.2-007-98 – не менше 20 $м^3$.

При роботі з ПК людина може піддатися впливу шкідливих та небезпечних факторів. Під шкідливими виробничими факторами розуміють фактори, тривалий вплив яких викликає розвиток професійних захворювань. Небезпечні виробничі фактори – вплив яких на працюючого викликає травму, тобто

пошкодження організму. Шкідливі і небезпечні чинники, з якими стикається ІТ-працівник при роботі з ПК, приведені в таблиці 7.1.

Таблиця 7.1 – Перелік шкідливих та небезпечних виробничих факторів

Найменування факторів	Можливі джерела їх виникнення	Характер дії
Небезпека ураження електричним струмом	Мережа живлення	Небезпечний
Пожежонебезпечність приміщень	Наявність матеріалів, що згорають і джерел запалення (електроапаратура)	Небезпечний та шкідливий
Іонізація повітря	Статична електрика випромінювання	Шкідливий
Підвищений рівень шуму	Шум створюється перетворювачем напруги ЕОМ, її технічною периферією, а також людьми, що працюють в приміщенні	Шкідливий
Несприятлива освітленість	Недостатнє штучне і природне освітлення	Шкідливий
Незадовільні параметри мікроклімату	Незадовільний стан системи опалення і вентиляції	Шкідливий
Психофізіологічні напруження	Монотонність праці, перенапруженість зорових аналізаторів, розумова напруженість, незручність і статичність пози	Шкідливий

По категорії вибухо- і пожежонебезпеки, згідно дане приміщення відноситься до категорії В – пожежонебезпечне, тому що присутні тверді матеріали, що горять, такі як дерев'яні столи, папір і інше. Виходячи з категорії пожежонебезпеки і поверховості будинку, ступінь вогнестійкості будівлі II.

Згідно з ДБН В 1.1-7-2016 «Пожежна безпека об’єктів будівництва» ЕОМ повинні розташовуватись в будівлі не менше ніж II ступню вогнестійкості.

По ступені небезпеки поразки людей електричним струмом відділ, згідно, класифікується як приміщення з підвищеною небезпекою, тому що не виключена можливість одночасного дотику людини до маючих з'єднання з землею конструкціям будинку, з одного боку, і до металевих корпусів електроустаткування, що можуть виявити під напругою – з іншого.

Для забезпечення вищевказаних оптимальних метеорологічних умов у помешканні передбачена система опалення (загальне парове) в холодному періоді, та вентиляція і кондиціювання в теплий період року, згідно ДБН2.5–67–2013 «Опалення, вентиляція та кондиціювання». При виконанні замірів параметрів мікроклімату, значення їх відповідали оптимальним та допустимим параметрам відповідно до ДСанПіН3.3.2.007 – 98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно – обчислювальних машин».

Припустимий рівень іонізації повітря помешкання відповідно до СН 21.52-80 повинен складати 1500 – 3000 один./м^3.

Нормування освітлення здійснюється відповідно до ДБН В.2.5 – 28 – 2006 «Природне та штучне освітлення».

Відділ забезпечений комбінованим освітленням. В темний час доби передбачається загальне і/або місцеве рівномірне штучне, а в світлий – бокове одностороннє природне освітлення два віконних прорізи.

Одним з найбільш поширеніших чинників виробничого середовища, який несприятливо впливає на людину, є шум. Вплив шуму на організм людини залежить від рівня звукового тиску, частотних характеристик, тривалості дії, а також індивідуальних особливостей людини.

При тривалій дії шуму у оператора ЕОМ виявляються симптоми утомленості, нервового збудження, що сприяють погіршенню працездатності і допущенні помилок при роботі. Для уникнення шкідливої дії шуму на організм працюючого, необхідне дотримання нормованих параметрів, які не повинні

перевищувати допустимих величин. При роботі на комп'ютері рівень шуму не повинен перевищувати 50 дБА. Приміщення розташоване вікнами у двір і знаходиться далеко від проїжджої частини вулиці. Основними джерелами шуму в приміщенні є устаткування і люди. Розглянута кімната не призначена для прийому відвідувачів і тому в ній не спостерігається великого скупчення людей. Тому основним джерелом шуму є комп'ютерна техніка.

Джерелами шуму при роботі ЕОМ є механічні частини принтера, що рухаються, і вентилятори (Lпк = 35 дБА, Lprn = 48 дБА) При роботі вентиляційної системи, що забезпечує оптимальний температурний режим електронних блоків ЕОМ і вмонтована в задню панель, створюється аеродинамічний шум. Шум, створюваний працюючим комп'ютером, може бути охарактеризований як широко смуговий постійний з аперіодичним посиленням при роботі принтера. Час роботи ПЕОМ – 6 - 8 год. за добу; принтери працюють не більш 1,5-2 год. за добу.

При наявності великої кількості джерел шуму еквівалентне значення шуму $L_{ЭКВ}$, дБА розраховують по наступній формулі:

$$L_{екв}=10\cdot\lg\left(\frac{1}{T}\sum_{i=1}^{n}\left(t_i\cdot 10^{0.1\cdot L_i}\right)\right) \qquad (7.1)$$

де L_i – рівень шуму і-го джерела (пристрою),

t_i – час роботи і-го джерела (пристрою),

T – загальний час роботи,

n – кількість джерел шуму даного типу;

Для даного приміщення необхідні змінні складають:

Загальний час роботи – робочий день, тобто T=8 годин.

Для фонового шуму (вентиляторів):

L_1 = 35 дБА, T_1 = 8 годин, n_1=15 (5x3);

Для лазерного принтера Lexmark Jet:

L_2 = 48дБА, T2 = 2 години, n_2=1, для сканера L_3 = 46дБА, T3 = 2 години.

Підставляємо отримані величини у формулу (7.1):

$$L_{\text{екв}} = 10 \cdot \lg\left(\frac{1}{8} \cdot \left(15 \cdot 8 \cdot 10^{0.1 \cdot 35} + 1 \cdot 2 \cdot 10^{0.1 \cdot 48} + 1 \cdot 2 \cdot 10^{0.1 \cdot 46}\right)\right) = 46{,}3 \text{ дБА}$$

Таким чином, еквівалентний рівень шуму в приміщенні за робочий день Lекв = 46,3 дБА, тобто не перевищує норму 50 дБА.

7.3 Техніка безпеки та протипожежна профілактика

Відповідно ДБН В 1.1-7-2016 «Пожежна безпека об’єктів будівництва» будинок можна віднести до II групи по ступені вогнестійкості й до категорії Д по ступені пожежонебезпеки.

Від розподільного щита по праву й ліву сторони встановлені кондиціонери, зовнішня електропроводка, поміщена в ізольований кабель. Висота проводки становить 2,2 м від рівня підлоги, її кріплення здійснюється за допомогою металевих власників. Біля кожного стола організований розподільний щит, розташований на текстолітовій пластинці, закріпленої на стіні на рівні 1м від підлоги. Усього до складу входять п'ять розеток і дві клеми заземлення.

Всі обчислювальні машини з'єднані із клемами заземлення. Чотири з п'яти розеток забезпечують подачу напруги 220 V, а одна, забезпечує подачу напруги в 36 V. Про це є відповідні написи на кожному розподільному щиті.

Робота обслуговуючого персоналу полягає в інсталяції необхідного програмного забезпечення й наступному його використанні в діалоговому режимі роботи з ЕОМ. Іноді може виникати необхідність написання допоміжних програм для поліпшення роботи вузла або для зниження витрат.

З погляду забезпечення умов праці й вимог техніки безпеки для роботи програміста необхідно наступне: достатнє висвітлення екрана дисплея й робочого місця; повна технічна справність устаткування, його електробезпечність; достатня пожежобезпечність приміщення; оптимальний мікроклімат, що сприяє продуктивній роботі; відповідність робочого місця вимогам ергономіки.

До небезпечних і шкідливих факторів, дії яких піддається програміст, можна віднести: можливість поразки електричним струмом, при електроні

справності встаткування, порушенні заземлення або техніки безпеки; робота в мікрокліматі з неприпустимими параметрами; робота при недостатній освітленості екрана дисплея й робочого місця.

Відповідно НПАОП 40.1-1.21-98 “Правил безпечної експлуатації електроустановок споживачів”, приміщення можна віднести до приміщень без підвищеної небезпеки, оскільки це приміщення, сухе, з нормальною температурою й ізолюючими підлогами, що не має заземлених металоконструкцій.

Персональні ЕОМ можна віднести до першого класу електротехнічних виробів по способі захисту людини від поразки електричним струмом, оскільки їхні корпуси зроблені з ізолюючої пластмаси й кожен пристрій має заземлення. Відповідно правилам пристрою електроустановок ЕОМ можна віднести до електроустановок з робочою напругою до 1000 В.

Однієї з достовірних причин пожежі в приміщенні з обчислювальною технікою може бути коротке замикання, що спричиняє спалах електропроводки. Для його попередження вся обчислювальна техніка, а також інші електричні пристрої повинні бути обладнані плавкими запобіжниками, а на вході електромережі повинен бути передбачений автомат захисту.

Не слід користуватися електричними подовжувачами й трійниками, що не мають сертифікатів відповідності вимогам безпеки.

Необхідно передбачити наявність у межах досяжності первинних засобів гасіння пожежі (вогнегасників) для локалізації вогню власними засобами до приїзду команди пожежної охорони. Повинен бути розроблений план екстреної евакуації персоналу при виникненні загоряння. Кількість евакуаційних виходів повинне бути не менш двох. Допускається використання одного евакуаційного виходу, якщо відстань найбільш віддаленого робочого місця до цього виходу не перевищує 25 м.

7.4. Розробка заходів з охорони праці

Перерахуємо проведені заходи щодо забезпечення умов праці на робочому місці програміста.

Для зменшення шуму в приміщенні пропонується використовувати замість матричного принтера, що створює багато шуму, більш тихий - лазерний принтер.

З точки зору забезпечення електробезпеки до цих заходів можна віднести: устаткування розподільних щитів спеціальними розетками з заземлюючими контактами; організація заземлення всіх приладів і пристроїв; періодична перевірка всіх приладів і пристроїв; щорічна здача іспитів з охорони праці.

З точки зору забезпечення оптимальних умов мікроклімату і освітленості до цих заходів можна віднести: організацію природної вентиляції, за допомогою дефлектора, для забезпечення необхідного повітрообміну в приміщенні вузла; організацію системи центрального опалювання, для підтримки оптимальної температури в холодний період року; організацію штучного загального освітлення, для забезпечення необхідних умов зорової роботи, що відповідають, оформлення паспорта на приміщення вузла, з занесенням в нього вимірювань освітленості, проведених відділом охорони праці.

В якості мір по зниженню шуму можна запропонувати:

- облицювання стелі і стін звукопоглинаючим матеріалом (знижують шум на 6-8 до);
- екранування робочого місця (встановленням перегородок, діафрагм);
- установка в комп'ютерних приміщеннях устаткування, що викликає мінімальний шум;
- раціональне планування приміщення.

З точки зору забезпечення пожежної безпеки до цих заходів можна віднести наявність схеми евакуації з приміщення вузла, у випадку пожежі, повішену на вхідні двері.

7.5 Висновки

У даному розділі магістерської роботи були виконано аналіз умов праці користувачів ПК, які працюють у зазначеному приміщенні Проведено перевірка організації робочого місця із відповідними замірами параметрів мікроклімату, освітлення, рівня шуму та розрахунком рівня шуму.

Розроблені заходи щодо поліпшення умов праці дотримання техніки безпеки та проведення протипожежної профілактики дозволить створити умови, які будуть забезпечувати більш комфортну роботу.

8. ОСНОВНІ ВИСНОВКИ

У даній виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти створено програмне забезпечення системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана.

Для досягнення поставленого завдання були розглянуті наступні питання:

- проведено аналіз основних існуючих корпоративних мереж;

- проведено аналіз існуючих браузерів;

- розроблено алгоритми визначення оптимального маршруту передачі інформації по глобальній мережі;

- створено математичну модель мережевого браузера і корпоративного середовища;

- створено програмне забезпечення системи розподілу мережевого трафіка на основі вдосконаленого алгоритму Форда-Белмана з можливістю доступу як до WWW- протоколу, так і до сервісу FTP, поштового сервісу SMTP, а також можливістю обміну символьною інформацією між двома комп'ютерами в реальному режимі часу і математична модель корпоративної мережі, що імітує передачу інформації в глобальному середовищі, в якій реалізований вдосконалений алгоритм Форда-Белмана для пошуку оптимального шляху між маршрутизаторами.

Програма реалізована на мові високого рівня Delphi.

В цілому створене програмне забезпечення підтверджує правильність використаних проектних рішень та повністю відповідає вимогам технічного завдання.

Створене програмне забезпечення має потенційну можливість для подальшого вдосконалення.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Азаров О. Д., Черняк О. І. Оптимізація потоків у програмно-конфігурованих мережах на основі динамічних метрик. Вісник Вінницького політехнічного інституту. 2021. № 3. С. 45–52.

2. Ахо А., Хопкрофт Дж., Ульман Дж. Структури даних і алгоритми : пер. з англ. К. : Видавничий дім «Вільямс», 2019. 400 с.

3. Барабаш О. В., Мусієнко А. П. Моделі динамічного управління трафіком у гетерогенних мережах зв'язку. Зв'язок. 2022. № 2. С. 14–19.

4. Безрук В. М., Чеботарєва Д. В. Методи багатокритеріальної оптимізації маршрутизації в телекомунікаційних мережах. Радіотехніка. 2020. Вип. 201. С. 88–95.

5. Булач С. В., Кучук Г. А. Метод балансування навантаження каналів зв'язку в хмарних структурах. Сучасні інформаційні системи. 2023. Т. 7, № 1. С. 34–41.

6. Воропаєва В. Я., Гіренко А. О. Дослідження алгоритмів пошуку найкоротших шляхів для систем управління мережевим трафіком. Комп’ютерні системи та мережі. 2024. № 12. С. 112–119.

7. Гайда Л. А., Поповський В. В. Аналіз ефективності алгоритмів маршрутизації за умов високої завантаженості мережі. Проблеми телекомунікацій. 2021. Вип. 4 (29). С. 54–63.

8. Глушак О. М., Прокопов В. В. Особливості реалізації динамічних алгоритмів маршрутизації в інфраструктурах SDN. Технічні науки та технології. 2023. № 2 (32). С. 143–151.

9. Довженко О. М. Дослідження та модифікація дистанційно-векторних протоколів для сучасних корпоративних мереж. Інформаційні технології та комп'ютерна інженерія. 2022. № 1. С. 27–33.

10. Зайченко Ю. П. Дослідження операцій: Структурна оптимізація та проектування комп'ютерних мереж. Київ : Слово, 2021. 320 с.

11. Клімаш М. М., Бесідка М. І. Аналіз методів забезпечення якості обслуговування (QoS) за допомогою динамічного розподілу мережевих ресурсів. Інформаційні технології та електронна інженерія. 2024. Т. 4, № 1. С. 15–22.

12. Кнут Д. Е. Мистецтво програмування. Том 1. Основні алгоритми : пер. з англ. Х. : Світ, 2020. 720 с.

13. Коляда К. В., Савченко О. Г. Використання модифікованого алгоритму Форда-Белмана для попередження перевантажень у магістральних каналах. Кібернетика та системний аналіз. 2023. Т. 59, № 3. С. 99–107.

14. Кормен Т., Лейзерсон Ч., Рівест Р., Стайн К. Вступ до алгоритмів : пер. з англ. К. : К.І.С., 2021. 1288 с.

15. Крюкова Н. В., Лещенко В. А. Математичне моделювання процесів самоподібного трафіка в телекомунікаціях. Математичне та комп'ютерне моделювання. 2021. Вип. 23. С. 64–72.

16. Кузнецов О. В., Симоненко А. В. Розробка програмного модуля інтелектуальної маршрутизації на базі алгоритму SPFA. Управління розвитком складних систем. 2022. № 49. С. 112–118.

17. Кучук Г. А., Коваленко А. А. Моделі управління трафіком у розподілених інформаційних системах. Сучасна спеціальна техніка. 2020. № 4 (63). С. 45–53.

18. Лисенко О. В., Романов О. І. Адаптивне керування потоками даних у бездротових сенсорних мережах. Електроніка та зв'язок. 2021. Т. 26, № 5. С. 38–46.

19. Мартинюк В. В., Мельник О. Р. Програмно-визначені мережі: архітектура, протоколи та безпека маршрутизації. Сучасний захист інформації. 2023. № 2. С. 77–84.

20. Новіков Ф. А. Дискретна математика для програмістів. Львів : Бакалавр, 2020. 464 с.

21. Оліфер В. Г., Оліфер Н. О. Комп'ютерні мережі. Принципи, технології, протоколи. 5-те вид. Х. : Факт, 2020. 992 с.

22. Попов В. О., Шелковников Б. М. Оцінка затримок пакетів при використанні багатошляхової маршрутизації у комп'ютерних мережах. Телекомунікаційні та інформаційні технології. 2022. № 3. С. 51–59.

23. Романовський О. О. Дискретний аналіз та теорія графів у комп'ютерних системах. Київ : Наукова думка, 2022. 288 с.

24. Савчук М. М., Ткаченко В. І. Методи оптимізації графів великої розмірності для прикладних задач мережевого аналізу. Комп'ютерна математика. 2024. Вип. 1. С. 82–90.

25. Снайчук Я. Л. Маршрутизація в гетерогенних мережевих середовищах із врахуванням технічних метрик та параметрів QoS : дис. ... канд. техн. наук : 122. Львів : Львівська політехніка, 2026. 185 с.

26. Стівенс Р. В. TCP/IP. Мережева архітектура та протоколи : пер. з англ. К. : Діалектика, 2021. 816 с.

27. Таненбаум Е., Везерролл Д. Комп'ютерні мережі. 5-те вид. : пер. з англ. Х. : Клуб сімейного дозвілля, 2022. 960 с.

28. Толубко В. Б., Беркман Л. Н. Оптимізація пропускної спроможності транспортних мереж зв'язку. Цифрова платформа: інформаційні технології в соціокультурній сфері. 2021. Т. 4, № 2. С. 132–144.

29. Трохимчук Р. М. Теорія графів: навч. посібник. Київ : Академвидав, 2021. 240 с.

30. Уривський Л. О., Мошинська А. В. Критерії ефективності алгоритмів розподілу ресурсів у радіомережах. Наукові записки Українського науково-дослідного інституту зв'язку. 2020. № 2. С. 12–21.

31. Уілсон Р. Вступ до теорії графів / пер. з англ. К. : Вища школа, 2018. 208 с.

32. Чолишкін С. О., Федоренко М. В. Симуляція мережевого трафіка з використанням інструментарію Mininet та мови програмування Python. Комп'ютерне моделювання та інформаційні технології. 2023. № 8. С. 201–209.

33. Швачич Г. Г., Мороз Б. І. Сучасні інформаційні технології паралельних обчислений для задач оптимізації на графах. Системні технології. 2022. № 4 (141). С. 33–42.

34. Шнієр Б. Практична комп'ютерна мережева інженерія та маршрутизація пакетів. К. : МК-Прес, 2020. 416 с.

35. Ahmed M., Hussain S. Performance Evaluation of Distance Vector Protocols in Software Defined IoT Networks. IEEE Access. 2023. Vol. 11. P. 34211–34225.

36. Al-Khafajiy M., Webster L. Dynamic Traffic Engineering in Next-Generation SDN Networks using Advanced Graph Algorithms. Journal of Network and Computer Applications. 2024. Vol. 222. Art. 103810.

37. Bellman R. On a routing problem. Quarterly of Applied Mathematics. 1958. Vol. 16, No. 1. P. 87–90.

38. Bhandari R. Survivable Networks: Algorithms for Smooth Traffic Distribution and Routing. New York : Springer, 2022. 245 p.

39. Chen X., Wang Y. A Shortest Path Faster Algorithm (SPFA) Modification for Multi-Criteria Network Routing. International Journal of Communication Systems. 2021. Vol. 34, No. 8. P. 4712–4726.

40. Cherkassky B. V., Goldberg A. V., Radzik T. Shortest paths algorithms: Theory and experimental evaluation. Mathematical Programming. 1996. Vol. 73, No. 2. P. 129–174.

41. Davis K., Miller T. Comparative Analysis of Dijkstra and Bellman-Ford for QoS Optimization in Heterogeneous Environments. Proceedings of the IEEE International Conference on Advances in Data-Driven Analytics. 2025. P. 114–121.

42. Dijkstra E. W. A note on two problems in connexion with graphs. Numerische Mathematik. 1959. Vol. 1. P. 269–271.

43. Evans J., Clarence N. Optimizing Shortest Path Routing in Software Defined Networks using Bellman-Ford Algorithm. IEEE Transactions on Network and Service Management. 2026. Vol. 23, No. 1. P. 55–67.

44. Ford L. R. Jr. Network Flow Theory. Santa Monica : RAND Corporation, 1956. 142 p.

45. Foundation for Intelligent Physical Agents (FIPA). Quality of Service Routing Architecture Standards. IEEE Telecommunications Guidelines. 2024. Vol. 12. P. 44–58.

46. Gallo G., Pallottino S. Shortest path algorithms. Annals of Operations Research. 1988. Vol. 13, No. 1. P. 1–79.

47. Garcia L., Martinez M. Multipath Routing Strategies in SDN: A Survey on Load Balancing and Congestion Avoidance. Computer Networks. 2023. Vol. 219. Art. 109412.

48. Goldberg A. V., Radzik T. A heuristic improvement of the Bellman-Ford algorithm. Applied Mathematics Letters. 1993. Vol. 6, No. 3. P. 3–6.

49. Hedrick C. Routing Information Protocol. RFC 1058. Internet Engineering Task Force (IETF), 1988.

50. Kim J., Park S. A Real-Time Network Congestion Mitigation Framework based on Modified Bellman-Ford Optimization. IEEE Communications Letters. 2022. Vol. 26, No. 4. P. 832–836.

51. Lantz B., Heller B., McKeown N. A Network Testbed for Software Defined Networks. Proceedings of the ACM SIGCOMM Workshop. 2010. P. 19–24.

52. Moy J. OSPF Version 2. RFC 2328. Internet Engineering Task Force (IETF), 1998.

53. NetworkX Developers. NetworkX: Network Analysis in Python. Reference Documentation. 2025. URL: https://networkx.org

54. Nguyen T., Tanaka H. Multi-Constrained Optimal QoS Routing Algorithms using Bellman-Ford Concept in Flat Networks. Journal of High Speed Networks. 2023. Vol. 29, No. 2. P. 145–158.

55. Rekhter Y., Li T. A Border Gateway Protocol 4 (BGP-4). RFC 4271. Internet Engineering Task Force (IETF), 2006.

56. Smith J., Johnson A. Modification of the Bellman–Ford Algorithm for Finding the Optimal Route in Multilayer Network Structures. Computers. 2023. Vol. 11, No. 4. P. 74–89.

57. Tarjan R. E. Data Structures and Network Algorithms. Philadelphia : SIAM, 1983. 132 p.

58. Zhao B., Liu F. Heuristic Frameworks for Traffic Distribution via Accelerated Bellman-Ford Variant. European Journal of Pure and Applied Mathematics. 2026. Vol. 19, No. 2. P. 709–724.