

Міністерство освіти і науки України  
Державний заклад  
«Луганський національний університет імені Тараса Шевченка»

Навчально-науковий інститут математики та інформаційних технологій

Кафедра інформаційних технологій та систем

**Люклянчук Павло Володимирович**

**АНАЛІЗ ТЕХНОЛОГІЙ ВІДДАЛЕНОГО ДОСТУПУ ТА РОЗРОБКА  
ПРОГРАМНОГО ДОДАТКУ ВІДДАЛЕНОГО КЕРУВАННЯ  
РОБОЧОЮ СТАНЦІЄЮ НА ПЛАТФОРМІ .NET**

**кваліфікаційна робота**

**здобувача вищої освіти другого (магістерського) рівня**

**освітньої програми «Комп'ютерні мережі»**

**за спеціальністю 123 Комп'ютерна інженерія**

Особистий підпис \_\_\_\_\_ Павло ЛЮКЛЯНЧУК

Науковий керівник \_\_\_\_\_ Володимир ДОНЧЕНКО,  
старший викладач  
кафедри інформаційних технологій  
та систем

В.о. завідувача кафедри \_\_\_\_\_ Микола СЕМЕНОВ,  
кандидат педагогічних наук, доцент  
кафедри інформаційних технологій  
та систем

Полтава – 2024

## АНОТАЦІЯ

**Люклянчук П. В.**

**Тема:** Аналіз технологій віддаленого доступу та розробка програмного додатку віддаленого керування робочою станцією на платформі .NET.

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

**Установа:** ЛНУ імені Тараса Шевченка, 2024р.

**Магістерська робота містить:** 67 с., 39 рис., 2 табл., 34 джерела.

**Об'єктом дослідження** є технології віддаленого доступу до ресурсів комп'ютера.

**Предметом дослідження** є програмний додаток віддаленого керування робочою станцією на платформі .NET.

**Мета роботи** - аналіз технологій віддаленого доступу та розробка програмного додатку віддаленого керування робочою станцією на платформі .NET..

**Методи дослідження:** *теоретичні:* аналіз наукової літератури, узагальнення та систематизація теоретичних положень про технології віддаленого доступу до ресурсів комп'ютеру; *емпіричні:* порівняльний аналіз додатків віддаленого доступу та можливостей інструментів розробки програмних додатків; *експериментальні:* тестування розробленої системи.

**Результати роботи** – проаналізовано технології віддаленого доступу та обґрунтовано вибір протоколів RDP і VNC для клієнтського додатку, розроблено програмний додаток віддаленого керування робочою станцією на платформі .NET.

**Ключові слова:** ВІДДАЛЕНИЙ ДОСТУП, ПРОТОКОЛ, ПЛАТФОРМА .NET, ТЕХНОЛОГІЯ.

## ANNOTATION

**Liuklianchuk Pavlo**

**Theme:** Analysis of remote access technologies and development of a software application for remote management of a workstation on the .NET platform.

**Speciality:** 123 " Computer Engineering ".

**Institution:** Luhansk Taras Shevchenko National University (LTSNU), 2024 year.

**Master's work of:** 67 p., 39 im, 34 sources.

**Object of research** is the technology of remote access to computer resources.

**Subject of research:** is the remote workstation management software application on the .NET platform..

**An aim of research is** - analysis of remote access technologies and development of a software application for remote management of a workstation on the .NET platform.

**Methods of research:** theoretical: analysis of scientific literature, generalization and systematization of theoretical positions on technologies of remote access to computer resources; empirical: comparative analysis of remote access applications and capabilities of software development tools; experimental: testing of the developed system.

**The results of the work** - analyzed the technologies remote access technologies were analyzed and the choice of RDP and VNC protocols for the client application was justified, a software application for remote control of a workstation on the .NET platform was developed.

**Keywords:** REMOTE ACCESS, PROTOCOL, .NET PLATFORM, TECHNOLOGY.

## **ЗМІСТ**

<b>ВСТУП.....</b>	<b>5</b>
<b>РОЗДІЛ 1. АНАЛІЗ ТЕХНОЛОГІЇ ВІДДАЛЕНОГО ДОСТУПУ ДО РЕСУРСІВ КОМП'ЮТЕРУ .....</b>	<b>7</b>
1.1. Поняття та принципи функціонування віддаленого доступу.....	7
1.2. Аналіз протоколів віддаленого доступу .....	10
1.3. Апаратне забезпечення для системи з віддаленим доступом .....	18
1.4. Аналіз додатків віддаленого доступу.....	26
Висновки до розділу .....	33
<b>РОЗДІЛ 2. РОЗРОБКА ПРОГРАМНОГО ДОДАТКУ ВІДДАЛЕНОГО КЕРУВАННЯ РОБОЧОЮ СТАНЦІЄЮ НА ПЛАТФОРМІ .NET .....</b>	<b>35</b>
2.1. Характеристика платформи .NET та обґрунтування інструментів розробки додатку .....	35
2.2. Розробка програмного додатку.....	40
2.3. Тестування програмного додатку.....	53
Висновок до розділу .....	61
<b>ВИСНОВКИ .....</b>	<b>63</b>
<b>СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ .....</b>	<b>65</b>
<b>ДОДАТОК.....</b>	<b>68</b>

## ВСТУП

В даний час можливість повноцінного доступу до віддаленого комп'ютера, що знаходиться в мережі інтернет або локальної мережі, є актуальною як серед домашніх користувачів, так і серед корпоративних. Застосування технології віддаленого робочого в даний момент дуже широке: допомога користувачам (технічна підтримка), доступ до робочого столу комп'ютера з дому (або навпаки), відрядження, спостереження за роботою користувачів (наприклад, вчитель у навчальному класі).

Визначення "віддалений доступ" охоплює різноманітні форми взаємодії між комп'ютерами, мережами та програмами. Це широке поняття включає безліч схем взаємодії, спільною рисою яких є використання глобальних каналів або мереж для забезпечення комунікації. Основною особливістю віддаленого доступу є несиметричність взаємодії, де на одному полюсі знаходиться центральна велика мережа або комп'ютер, а на іншому – окремий віддалений термінал, комп'ютер або невелика мережа. Останні мають за мету отримати доступ до інформаційних ресурсів центральної мережі.

Віддалений доступ може бути здійснений в межах однієї операційної системи або між різними платформами на клієнтському та серверному рівнях. Це дає користувачу можливість працювати з комп'ютером на відстані так, ніби він управляє ним через локально підключений термінал. У цьому режимі він може запускати програми на віддаленому комп'ютері та спостерігати за результатами їх виконання. Тому саме розробка програмного додатку віддаленого керування робочою станцією стала основною метою магістерської роботи.

**Метою** магістерської роботи є аналіз технологій віддаленого доступу та розробка програмного додатку віддаленого керування робочою станцією на платформі .NET.

**Об'єктом** дослідження є технології віддаленого доступу до ресурсів комп'ютера.

**Предметом дослідження** є програмний додаток віддаленого керування робочою станцією на платформі .NET.

Відповідно до предмета, мети було визначено основні **завдання дослідження**:

- дослідження поняття та принципів функціонування віддаленого доступу;
- аналіз протоколів віддаленого доступу;
- аналіз додатків віддаленого доступу;
- огляд платформи .NET та обґрунтування інструментів розробки додатку;
- розробка програмного додатку віддаленого керування робочою станцією на платформі .NET.

Для вирішення завдань дослідження використано такі **методи дослідження**: *теоретичні*: аналіз наукової літератури, узагальнення та систематизація теоретичних положень про технології віддаленого доступу до ресурсів комп'ютеру; *емпіричні*: порівняний аналіз додатків віддаленого доступу та можливостей інструментів розробки програмних додатків; *експериментальні*: тестування розробленої системи.

До складу роботи входять два розділи, в яких проаналізовано технології віддаленого доступу та обґрунтовано вибір протоколів RDP і VNC для клієнтського додатку.

Практичним результатом роботи є розроблений клієнтський додаток віддаленого доступу на платформі .NET.

## РОЗДІЛ 1. АНАЛІЗ ТЕХНОЛОГІЇ ВІДДАЛЕНОГО ДОСТУПУ ДО РЕСУРСІВ КОМП'ЮТЕРУ

### 1.1. Поняття та принципи функціонування віддаленого доступу

*Віддалений доступ* являє собою функцію, що дозволяє користувачеві підключатися до комп'ютера через Інтернет за допомогою іншого ПК. Умовою для застосування такої опції є включений комп'ютер, до якого потрібно підключитися, а також встановлена і запущена функція віддаленого доступу. Здійснити таке з'єднання можна за допомогою будь-якого ПК, приєднаного до мережі [3].

Опція відкриває можливість користувачу використовувати свій комп'ютер віддалено, і забезпечує наступні додаткові можливості:

- Повний або частковий доступ до робочого столу, дисків і файлів користувача. Додатково налаштовується можливість виконання операцій видалення, копіювання і т. д.
- Організація голосового чату при наявності відповідного обладнання. Немає необхідності в додатковому телефонному зв'язку.
- Обмін файлами будь-якого формату без обмеження в обсязі.
- Запуск презентаційних матеріалів в режимі конференції.
- Відстеження і запис використання комп'ютера користувачами.
- Збір інформації про відвідані сайти, вхідні та вихідні електронні листи, використані месенджери.

Існують кілька видів віддаленого доступу:

- комп'ютер-мережа, що дозволяє контролювати роботу локальної мережі офісу або інтернет-кафе;
- термінал-комп'ютер, який спрощує зв'язок користувача з системою, наприклад, платіжні термінали банків;
- комп'ютер-комп'ютер, встановлює зв'язок між двома віддаленими комп'ютерами;
- мережа-мережа, відмінний інструмент при необхідності взаємодії

між віддаленими корпоративними мережами.

Одним з видів віддаленого доступу є доступ до *віддаленого робочого столу (Remote Desktop)* - це термін, яким позначається режим управління, коли один комп'ютер отримує права адміністратора по відношенню до іншого, віддаленого. Зв'язок між пристроями відбувається в реальному часі за допомогою Інтернет або локальної мережі.

Рівень доступу в режимі віддаленого адміністрування визначається конкретними завданнями і може бути змінений за потребою, наприклад:

- в одному випадку, підключення до робочої сесії дає можливість повного контролю і взаємодії з віддаленим комп'ютером, при якому допускається запуск на ньому програм і маніпуляції з файлами;
- в іншому, віддалений доступ до робочого столу дозволяє лише вести спостереження за процесами, без втручання в роботу його системи.

Доступ до віддаленого робочого столу здійснюється на базі архітектури «сервер - термінал», що дозволяє працювати з ресурсами сервера так само, як і з локальними ресурсами: виконувати команди, працювати з файловою системою, запускати додатки і т. д.

Початково, термінал - це кінцевий мережевий пристрій, підключений до обчислювальної системи і призначений для введення і виведення даних. Команди, що приймаються з пристроєм введення терміналу (клавіатури), передаються на віддалений сервер, де і виконуються. Результати обробки повертаються і відображаються на пристрої виведення терміналу (дисплеї). Згодом були розроблені емулятори терміналів - спеціальні програми, що виконують ті ж завдання.

Таким чином, можна виділити два основних типи терміналів:

Реальний, або фізичний термінал - як правило має на увазі пристрій, обчислювальні можливості якого обмежені можливістю відображати те, що йому передано з мережі (як максимум - повноекранну графіком).

Віртуальний термінал - мережевий додаток (програма), що виконує функції фізичного терміналу. З боку віддаленого хоста така програма, нічим не відрізняється від реального терміналу.

Термінальний доступ - це можливість отримання доступу до інформаційної системи або персонального комп'ютера таким чином, що локальна термінальна машина не виконує обчислювальні завдання, а лише направляє введену інформацію (з миші та клавіатури) на центральну машину (термінальний сервер), а також відображає графічну інформацію на моніторі (рис. 1.1). Важливо відзначити, що всі обчислювальні операції в термінальній системі виконуються на центральній машині.



Рис. 1.1. Технологія термінального доступу

Віддалене адміністрування - попередньо встановлена функція практично в кожній відомій сьогодні операційній системі, одночасно з цим, існує досить велика кількість програм, які роблять цей процес більш зручним, додають в стандартні версії нові функції.

Так, в операційній системі Windows вбудована можливість використання віддаленого робочого столу із застосуванням статичних IP-адресів на віддаленому комп'ютері. Налаштування віддаленого робочого столу (рис. 1.2) потребує прав адміністратора.

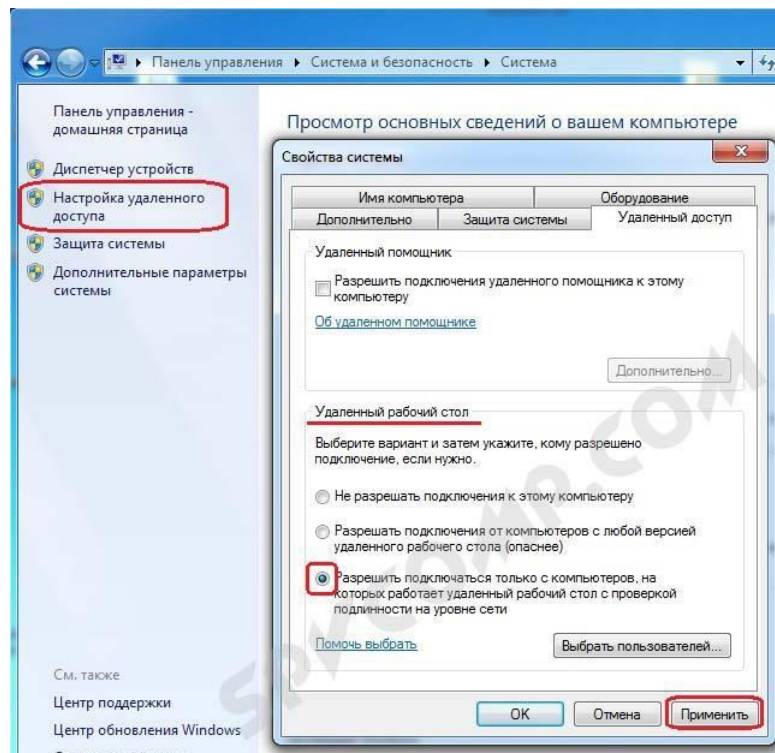


Рис. 1.2. Приклад вікна налагодження віддаленого робочого столу

Таким чином, застосування можливостей віддаленого доступу дає більш широкі можливості в корпоративному використанні ресурсів персонального комп'ютеру.

## 1.2. Аналіз протоколів віддаленого доступу

Принцип роботи протоколів віддаленого доступу полягає в наступному: для виконання програм на сервері емулюється пристрій виведення зображення (GDI). Програма виконує звичайний вивід на екран, а протоколи упаковують ці дані у зручний для передачі по мережі формат, і передають клієнту. Клієнт здійснює форматування команд і виконує на локальному екрані користувача. У зворотному порядку всі дії користувача (рухи миші, натискання клавіш) передаються від програми на тонкому клієнті до сервера, де відповідний драйвер імітує ситуацію як би дії приходили з локальної миші та клавіатури.

### Протокол RDP

RDP (англ. Remote Desktop Protocol) – це протокол на рівні застосунків, який використовується для організації віддаленої роботи користувача з сервером, на якому запущений сервіс термінальних підключень Microsoft Windows Remote Desktop Services CAL (RDS, попередня назва Terminal

Services) [21]. Клієнтські програми існують практично для всіх версій Windows (включаючи Windows CE та Mobile), Linux, FreeBSD, Mac OS X [5]. Протокол RDP реалізований у вигляді COM-компонента (Component Object Model – модель компонентних об'єктів) Microsoft RDP Client Control, який поставляється в складі операційної системи MS Windows. Протокол є прикладним, що базується на TCP. За замовчуванням використовується порт TCP 3389.

Microsoft передбачає два режими використання протоколу RDP:

- для адміністрування (режим віддаленого адміністрування). RDP в режимі адміністрування, використовується всіма сучасними операційними системами Microsoft Windows;
- для доступу до терміналів сервера (режим TerminalServer) [8].

Режим доступу до сервера терміналів можливий тільки в серверних версіях Windows.

Серверні варіанти операційної системи Windows дозволяють одночасно здійснювати два віддалених підключення та один локальний вхід в систему, у той час як клієнтські версії підтримують лише один вхід (будь то локальний або віддалений). Можливість віддаленого доступу до сервера терміналів стає доступною лише після встановлення відповідних ліцензій на ліцензійний сервер (рис. 1.3).



Рис. 1.3. RDP в режимі доступу до сервера терміналів

При використанні кластера термінальних серверів і балансування навантаження потрібна установка спеціалізованого сервера підключень (Session Directory Service). Даний сервер індексує сесії користувачів, що дозволяє здійснювати вхід та повторний вхід на термінальні сервери, які функціонують у розподіленому середовищі. Після встановлення з'єднання на транспортному рівні ініціалізується RDP-сесія, в межах якої узгоджуються різні параметри передачі даних. Передача виводу за допомогою примітивів є високо-пріоритетною функцією протоколу RDP, що значно економить трафік. Зображення передається тільки у випадку, якщо це абсолютно необхідно з яких-небудь причин (наприклад, у випадку неможливості узгодження параметрів передачі примітивів при налаштуванні RDP-сесії)[21]. Отримані команди обробляються RDP-клієнтом, який виводить зображення за допомогою своєї графічної підсистеми (див. рис. 1.4).

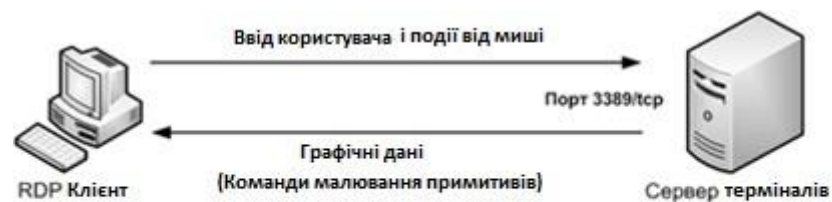


Рис. 1.4. Принцип роботи RDP

Протокол RDP дозволяє використовувати кілька віртуальних каналів у межах одного з'єднання для надання додаткового функціоналу:

- використання принтера або послідовного порту;
- перенаправлення файлової системи;
- підтримка роботи з буфером обміну;
- використання аудіо-підсистеми.

Параметри віртуальних каналів узгоджуються під час встановлення з'єднання.

Робота протоколу при стандартних налаштуваннях стиснення і глибині кольору 32 біт займає максимум 160 Кбайт/с ширини пропускання каналу, і дана цифра зменшується зі зниженням глибини кольору. При глибині кольору

8 біт для роботи протоколу необхідно максимум 60 Кбайт /с [6].

*Забезпечення безпеки при використанні RDP* (рис.1.5). Специфікація протоколу RDP передбачає використання одного із двох підходів для забезпечення безпеки:

- Standard RDP Security (вбудована підсистема безпеки);
- Enhanced RDP Security (зовнішня підсистема безпеки).



Рис. 1.5. Забезпечення безпеки при використанні RDP

*Standard RDP Security.* При цьому підході аутентифікація, шифрування і забезпечення цілісності реалізується засобами, закладеними в RDP протокол. Розглянемо ці етапи докладніше.

Аутентифікація сервера виконується наступним чином:

1. при старті системи генерується пара RSA ключів;
2. створюється сертифікат (Proprietary Certificate) відкритого ключа;
3. сертифікат підписується RSA-ключем, вбудованим в операційну систему (будь RDP клієнт містить відкритий ключ даного вбудованого RSA- ключа);
4. клієнт підключається до сервера терміналів і отримує підписаний сертифікат;
5. клієнт проводить аутентифікацію сертифіката та отримує відкритий ключ сервера, який подальше використовується для узгодження параметрів шифрування.

Клієнтська аутентифікація здійснюється шляхом введення імені користувача та пароля.

Для шифрування використовується потоковий шифр RC4 залежно від версії операційної системи, де доступні різні довжини ключа від 40 до 168 біт. Під час встановлення з'єднання, після узгодження довжини ключа, генеруються два унікальні ключі: один для шифрування даних від клієнта та інший - від сервера.

Цілісність повідомлення забезпечується за допомогою алгоритму генерації коду автентифікації повідомлення (MAC) на основі алгоритмів MD5 і SHA1.

Починаючи з Windows 2003 Server, для забезпечення сумісності з вимогами стандарту FIPS (Federal Information Processing Standard) 140-1 можливе використання алгоритму DES для шифрування повідомлень і алгоритму генерації MAC, що використовує тільки SHA1, для забезпечення цілісності.

*Enhanced RDP Security.* В даному підході використовуються зовнішні модулі забезпечення безпеки: TLS 1.0 і CredSSP [5].

Протокол TLS можна використовувати, починаючи з версії Windows 2003 Server, але тільки якщо його підтримує RDP-клієнт. Підтримка TLS додана, починаючи з RDP-клієнта версії 6.0

При використанні TLS, можна створити сертифікат сервера за допомогою Terminal Services або вибрати наявний сертифікат зі сховища Windows. Протокол CredSSP поєднує в собі можливості TLS, Kerberos і NTLM.

Розглянемо основні переваги протоколу CredSSP [5]:

- перевірка дозволу на вхід в віддалену систему до установки повноцінного RDP-з'єднання, що дозволяє економити ресурси сервера терміналів при великій кількості підключень;
- надійна аутентифікація і шифрування по протоколу TLS;
- використання одноразового входу в систему (Single Sign On) за допомогою Kerberos або NTLM.

## Протокол VNC

VNC (англ. Virtual Network Computing) - система віддаленого доступу, що базується на концепції RFB (англ. Remote FrameBuffer, віддалений кадровий буфер) [8] (рис. 1.6).

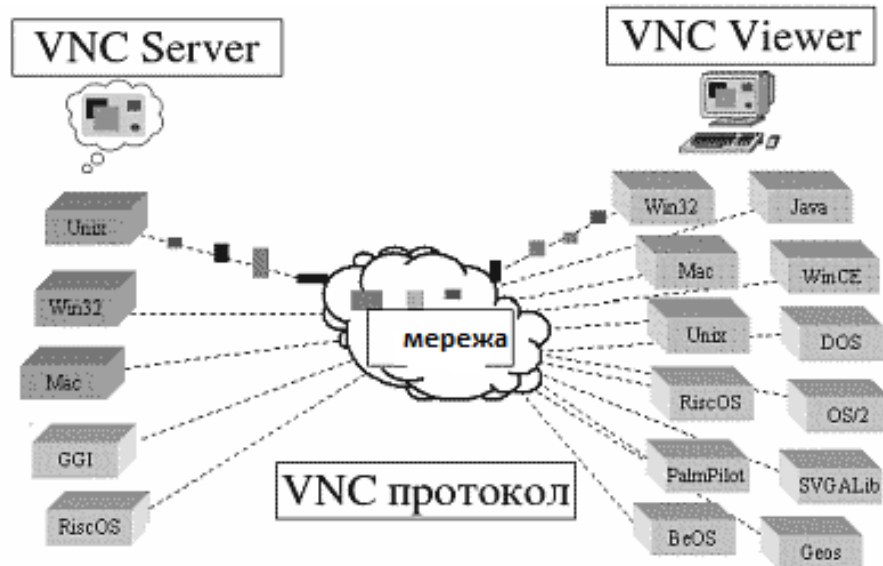


Рис. 1.6. VNC система віддаленого доступу

Система VNC є незалежною від платформи: VNC-клієнт, відомий як VNC viewer, запущений на одній операційній системі, може здійснювати з'єднання з VNC-сервером, який працює на будь-якій іншій операційній системі. У протоколі VNC реалізований наступний принцип роботи: дані про натискання клавіш і рух миші, що виконуються на власному комп'ютері, передаються по мережі на віддалений комп'ютер і сприймаються ним як дії з його власними клавіатурою і мишкою. Схема виведення з віддаленого комп'ютера на екран користувача ґрунтується на графічних примітивах (прямокутник піксельних даних виводиться в заданій координатами позиції) в своїй примітивній формі споживає більшу частину пропускнуєї можливості каналу. Існують різні кодування - методи визначення найбільш ефективного способу передачі цих прямокутників. Протокол VNC дозволяє клієнту і серверу «домовитися» про те, яке кодування буде використане.

Найпростіший метод кодування, що підтримується всіма клієнтами і серверами - «raw encoding», при якому пікселі передаються в порядку зліва-

направо, зверху-вниз, і після передачі початкового стану екрану передаються тільки пікселі, що змінилися. Цей метод працює дуже добре при незначних змінах зображення на екрані (руху покажчика миші по робочому столу, набір тексту під курсором), але завантаження каналу стає дуже високою при одночасному зміні великої кількості пікселів, наприклад, при перегляді відео в повноекранному режимі.

Для роботи потрібна ширина пропускання каналу становить від 32 Кбіт/с до 2 Мбіт/с. Для комфортної роботи в кольоровому режимі при дозволі екрану 1024x768 швидкість каналу повинна бути 1-2Мбіт/с.

Канал використовується повністю тільки при оновленні великих ділянок екрану, при друку тексту трафік помітно менше, а в решту часу канал практично не використовується. Якщо при передачі по каналу виникають великі затримки передачі пакетів (повільні канали, супутниковий зв'язок, великі відстані), це викликає погіршення часу реакції на натискання клавіш і рух миші, що значно знижує комфортність роботи.

*Забезпечення безпеки при використанні VNC.* Метод забезпечення безпеки (security type) визначається на стадії «рукоштовування», після узгодження версії протоколу, який буде використовуватися. Специфікація протоколу описує наступні методи забезпечення безпеки:

- відсутність аутентифікації і шифрування трафіку;
- аутентифікація в VNC не використовує шифрування трафіку, але пароль не передається у відкритому вигляді;
- використовується алгоритм "виклик-відповідь" з DES-шифруванням (ефективна довжина ключа становить 56 біт).

Більше того, багато сучасних реалізацій VNC підтримують розширення стандартного протоколу, які забезпечують шифрування і/або стиснення трафіку VNC, визначення доступу за списками ACL та різні методи аутентифікації.

Даним протоколом надаються такі технології і можливості [9]:

- обмін файлами;
- вагоме збільшення швидкості відтворення екрану віддаленого

- комп'ютера, в локальній мережі;
- підключення плагіна шифрування для поліпшення безпечного обміну даними;
- підтримка доменної аутентифікації;
- підтримка чату з віддаленим комп'ютером для обміну повідомленнями;
- ViewerToolbar, JavaViewer з підтримкою передачі файлів;
- авто масштабування для підгонки розміру зображення віддаленого комп'ютера;
- підтримка декількох моніторів;
- Repeater / Proxy-support;
- друк файлів з VNC-сервера на принтер за замовчуванням, підключений до комп'ютера VNC клієнта;
- можливість підключення VNC клієнта через різні web-проксі сервера і фільтри, що робить його використання таким же простим як і використання браузера.

### **Протокол ICA**

ICA (Independent Computing Architecture) - це приватний протокол, розроблений компанією Citrix Systems для сервера додатків. Цей протокол визначає специфікацію обміну даними між сервером і клієнтами, але не вбудований у жодну з платформ. ICA виконує завдання, які в багатьох відношеннях схожі на X Window System [10].

Протокол ICA забезпечує стандартну підтримку графічного виведення і введення даних з клавіатури, миші. Базовий механізм передачі між сервером і клієнтом графічних даних і даних, що вводяться з клавіатури / миші, по протоколу ICA майже ідентичний протоколу RDP.

*Забезпечення безпеки при використанні ICA.* Так як специфікація протоколу є закритою, слід говорити про шифрування в продуктах на основі даного протоколу. Наприклад, Citrix Metaframe поставляється тільки з мінімальними функціями шифрування для ICA-підключень і підтримує тільки

режим найпростішого шифрування, який задіює простий експортований алгоритм з менш ніж 40-розрядним ключем. Якщо необхідно організувати захист для підключень ICA, то слід придбати додатковий пакет Secure ICA для Meta Frame. У Secure ICA використаний алгоритм шифрування RSA RC5 і підтримуються 40-, 56- і 128-розрядні ключі.

Для шифрування комунікація між сервером і клієнтом ICA може використовуватися CitrixSSL Relay, який забезпечує шифрування по протоколу Secure Sockets Layer / Transport LayerSecurity (SSL / TLS) [10].

На підставі проведеного аналізу було вирішено обрати для розробки проекту протоколи RDP і VNC, оскільки їх застосування в одному клієнтському додатку дозволить реалізувати віддалений доступ не тільки до комп'ютерів з операційною системою Windows, а також дасть можливість застосовувати різні платформи на клієнті і сервері віддаленого доступу. Крім того, дані протоколи забезпечують високий рівень безпеки, також для цих протоколів існує доступний інтерфейс програмування, що спрощує розробку клієнтських додатків.

### **1.3. Апаратне забезпечення для системи з віддаленим доступом**

Одним з важливіших етапів створення програмного додатку віддаленого керування робочою станцією є вибір засобу приєднання локальної мережі до мережі Інтернет.

Існує кілька методів, але всі вони об'єднуються у два поширені підходи:

1. Створення порогового пристрою на основі виділеної обчислювальної машини із кількома мережевими адаптерами, а також налаштування системи доступу за допомогою можливостей конкретної операційної системи.
2. Використання окремого роутера в якості порогового пристрою та його налаштування.

В межах цієї роботи розглянемо другий варіант. Безумовно, існує велика кількість роутерів, які можуть бути використані в якості порогового обладнання.

В межах цієї роботи розглянемо тільки деякі приклади:

- WI-FI роутер Tp\_link TL-WR840N [11];
- WI-FI роутер Mercusys AC12g[12];
- WI-FI роутер Tp\_link AX1500 Wi-Fi 6[13];
- Роутер MikroTik RB750Gr3 без підтримки WI-FI[14]

### WI-FI роутер Tp\_link TL-WR840N

Це недорогий роутер швидкістю до 300 МБ/с, який має 4 LAN порти зі швидкістю 100МБ/с та 1 WAN порт та відносно не дорогий – 700 грн, кількість антен – 2.

Для його налаштування використовується браузер, адреса за замовчанням 192.168.0.1 (на рис 1.7 адреса вже налаштована на 192.168.112.1).

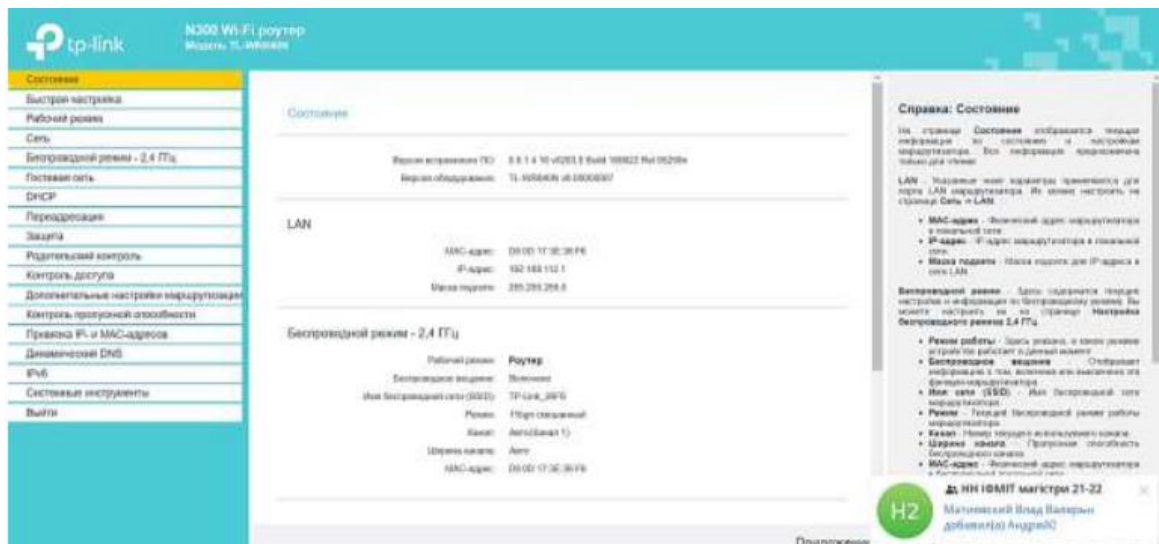


Рис. 1.7. Стан Tp\_link TL-WR840N

Для його попереднього налаштування слід задати параметри зовнішнього підключення, локальної мережі та параметри DHCP серверу (рис.1.8).

### WI-FI роутер Mercusys AC12g

Це більш сучасний роутер має загальну швидкість до 1200 Мбіт/с у двох діапазонах, який має 3 LAN порти зі швидкістю 1ГБ/с та 1 WAN порт та відносно не дорогий – 1500 грн.



Рис. 1.8. Налаштування та підготовка до роботи

Для його налаштування використовується браузер, адреса за замовчанням 192.168.0.1 (на рис 1.9 адреса вже налаштована на 192.168.113.1).

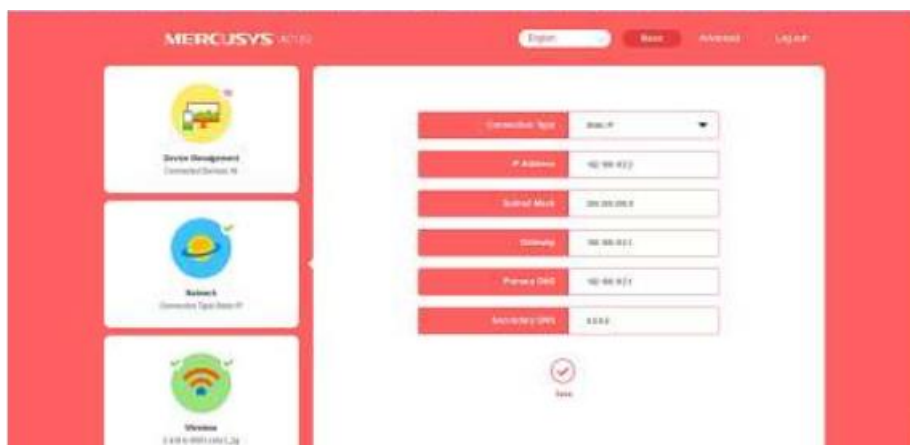


Рис. 1.9. Стан роутера Mercusys AC12g

Для його попереднього налаштування слід задати параметри зовнішнього підключення, локальної мережі та параметри DHCP серверу (рис.1.10-1.12).



Рис. 1.10. Зовнішня адреса



Рис. 1.11. Локальна адреса

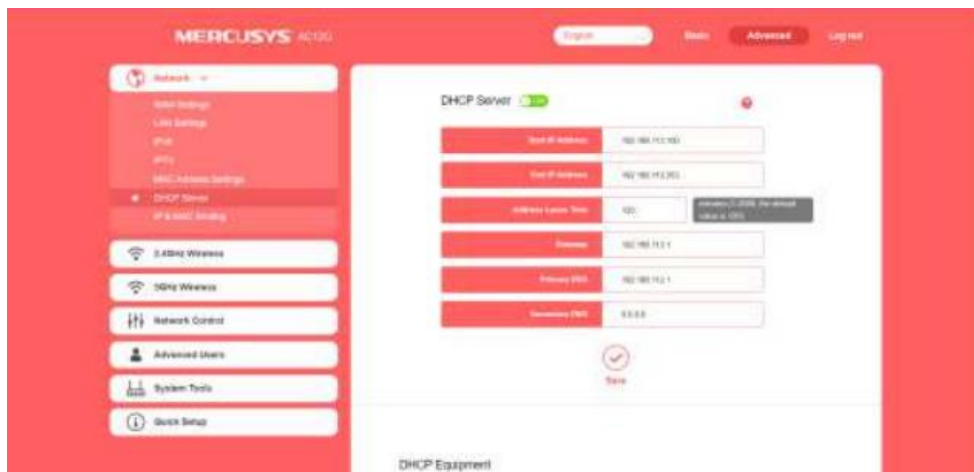


Рис. 1.12. Налаштування DHCP WI-FI роутера Mercusys AC12g  
**AX1500 Wi-Fi 6 Router**

Це сучасний роутер має загальну швидкість до 1500 Мбіт/с у двох діапазонах, який має 4 LAN порти зі швидкістю 1ГБ/с та 1 WAN порт та відносно не дорогий – 2500 грн. Для його налаштування використовується браузер, адреса за замовчанням 192.168.0.1 (на рис 1.13).

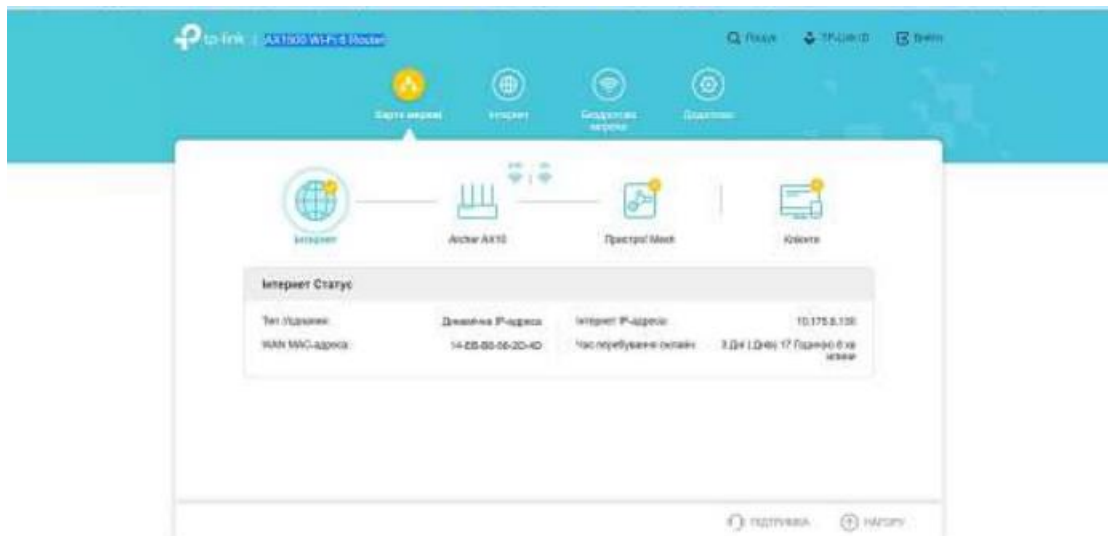


Рис. 1.13 Стан роутера

Всі налаштування робляться аналогічно попереднім роутерам. Слід задати параметри зовнішнього підключення, локальної мережі та параметри DHCP серверу.

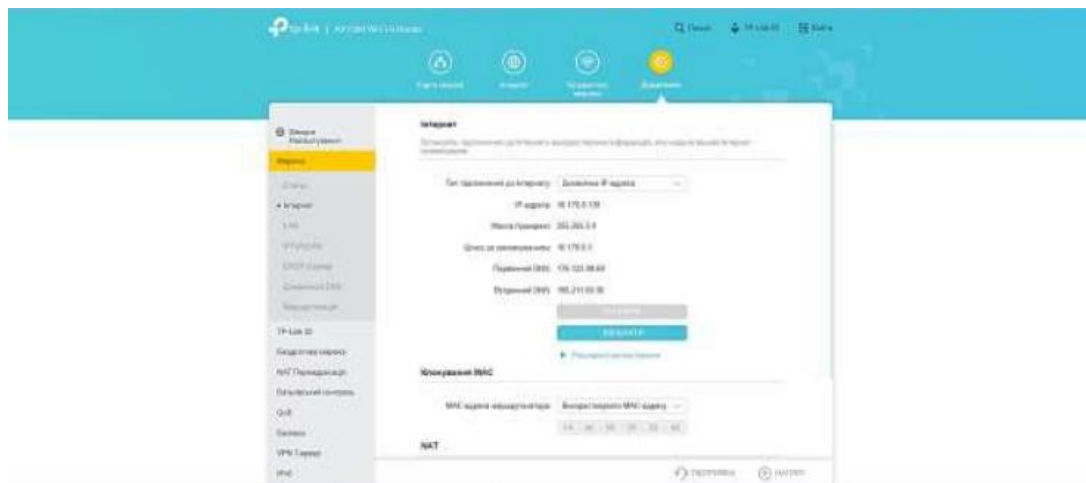


Рис. 1.14.а. Налаштування зовнішньої адреси



Рис. 1.14.б. Налаштування зовнішньої адреси



Рис. 1.15. Налаштування DHCP

## Роутер MikroTik RB750Gr3

Налаштування цього роутера суттєво відрізняється від попередніх. Роутер не має попередньо призначених інтернет портів та локальних портів.

Роутер має 5 рівноцінних портів зі швидкістю 1ГБ/с та ціну 2400 грн. Для налаштування використовується браузер або спеціалізована програма WINBOX. Після очищення конфігурації (або за замовчанням) вважається, що 1 порт (ether1) – Інтернет з'єднання, а з 2 по 5 – локальна мережа. [15] Слід відзначити, що роутери бренду MikroTik мають уніфікований принцип налаштування (внутрішня операційна система RouterOS) та відрізняються тільки характеристиками суцього технічного характеру: кількість портів, типи портів, швидкість маршрутизації пакетів, обсяг таблиці MAC адрес та інше. У цьому дослідженні ми використовуємо роутер MikroTik RB750Gr3 з операційною системою RouterOS v6.49.6 (стабільна версія) як пороговий пристрій. Для забезпечення достовірності даних ми виконуємо скидання конфігурації MikroTik (рис. 1.16. а) та знову налаштовуємо її (як показано на рис. 1.16. б).

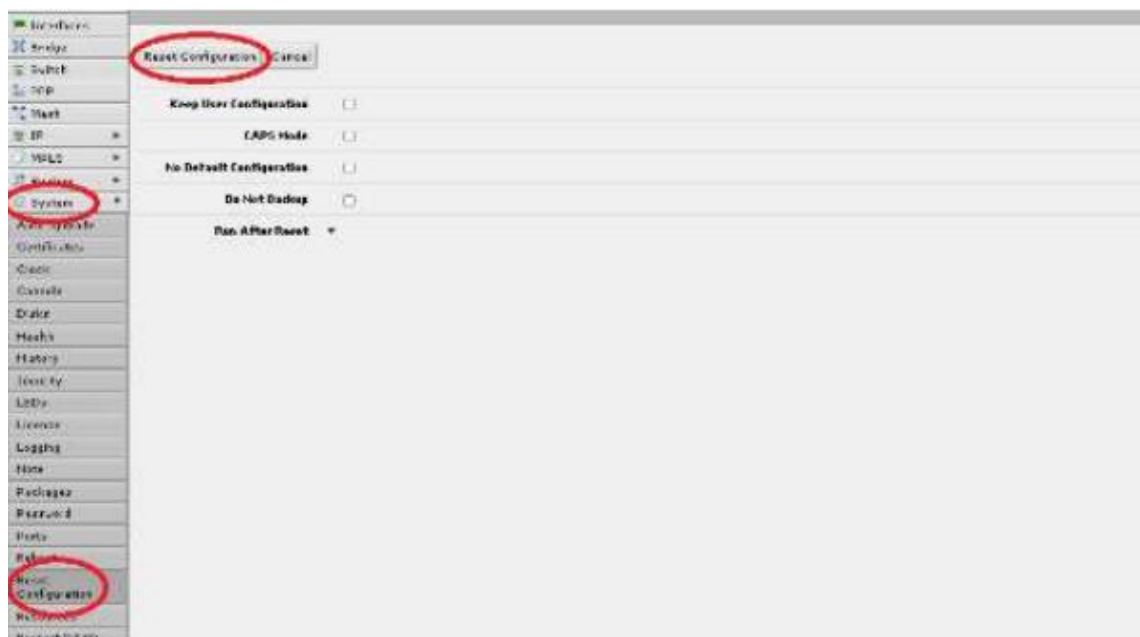


Рис. 1.16. а. Скидання конфігурації та попереднє налаштування

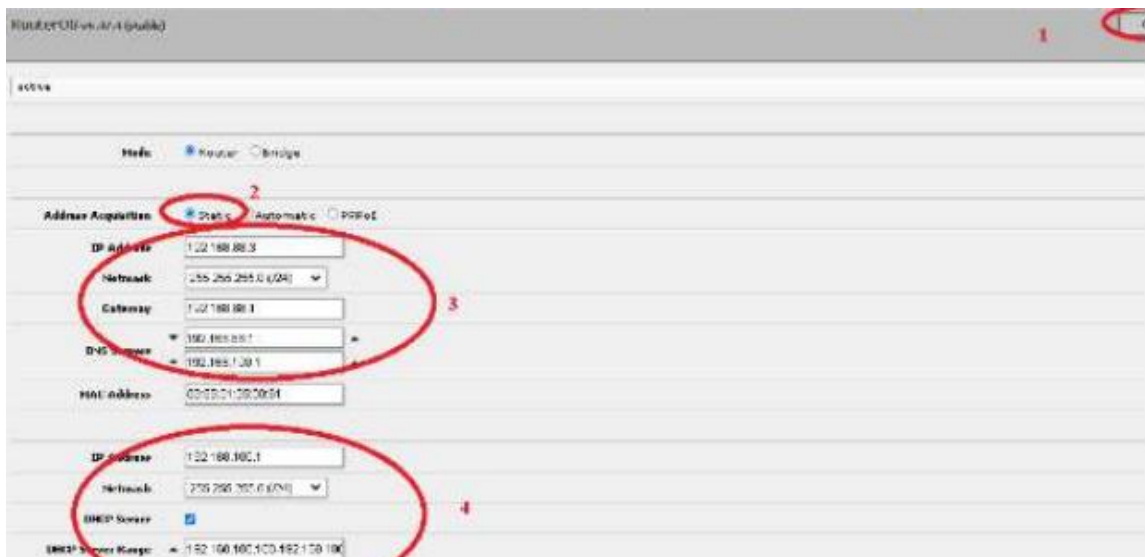


Рис. 1.16. б. Скидання конфігурації та попереднє налаштування

Отже, припустимо, що внутрішня локальна мережа має адресу 192.168.100.0 з маскою 255.255.255.0. Внутрішня адреса порогового пристрою (роутера MikroTik) – 192.168.100.1. Зовнішній інтерфейс, підключений до Інтернету через перший порт (ether1), має реальну IP-адресу, наприклад, 91.222.42.145, яка отримана від провайдера за допомогою NAT на адресу 192.168.88.3 (рис. 1.17).

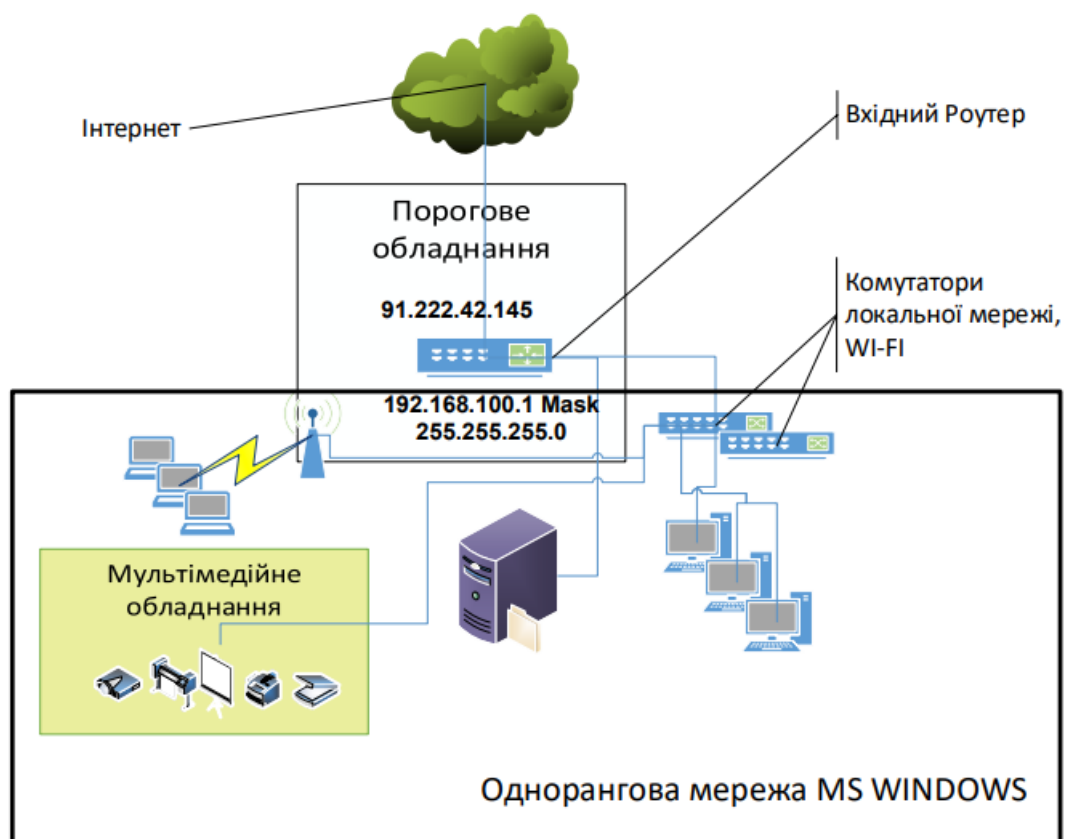


Рис. 1.17. Приклад адресації мережі

Один із способів перевірити зовнішні налаштування роутера MikroTik полягає в переході з локальної мережі за адресою порогового роутера, у даному випадку: <http://192.168.100.1>, та виборі меню "IP" – "Cloud", а потім натисканні кнопки "Force Update". Після цього слід зачекати, доки відбудеться оновлення (рис. 1.18.a) і переглянути отримане повідомлення у верхній частині вікна додатку (рис. 1.18.б).

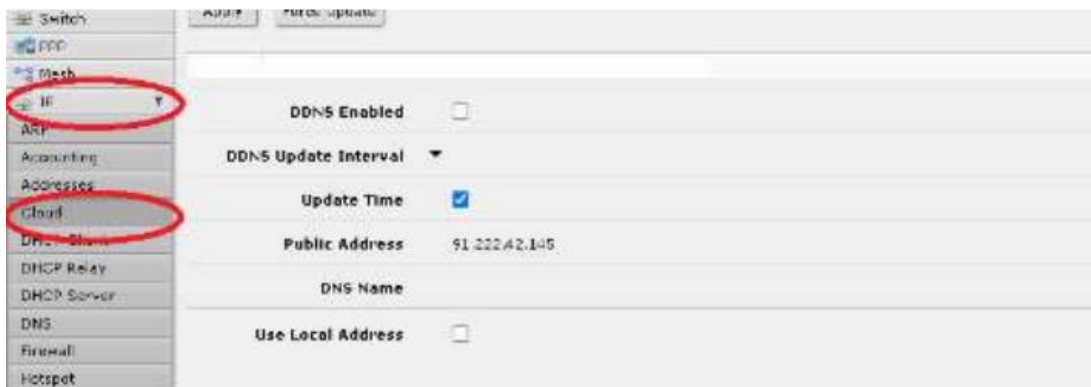


Рис. 1.18.a. Перевірка порогового роутера

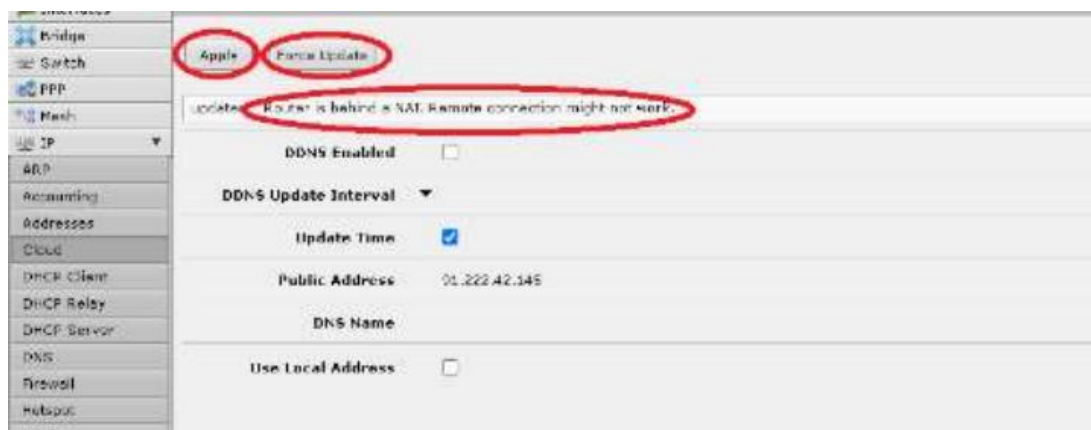


Рис. 1.18.б. Перевірка порогового роутера

Якщо ви отримали повідомлення, подібне до того, що показано на рис. 1.18.б: "Router is behind a NAT. Remote connection might not work." або "...service might not work", це може свідчити про відсутність у вас реальної IP-адреси. У такому випадку вам слід звернутися до свого провайдера. В ході практичного впровадження було виявлено, що оновлення працює нестійко із значними затримками, і у деяких випадках може не виводити помилки, навіть якщо відсутня реальна IP-адреса.

#### 1.4. Аналіз додатків віддаленого доступу

В даний час існує величезна кількість різних інструментів, які дозволяють дистанційно керувати комп'ютером або сервером. Кожен з них має свої плюси і мінуси, різні функціональні можливості, платні або безкоштовні версії, а також мобільні або тільки десктопні варіанти.

Проведемо порівняльний аналіз найбільш популярних програм віддаленого доступу, - *TeamViewer*, *LiteManager*, *Ammy admin*, *RAdmin*, виділимо їх переваги та недоліки. Аналіз основних порівняльних характеристик подано у таблиці 1.1.

### **TeamViewer**

Одна з найпопулярніших програм для віддаленого доступу, її можна швидко завантажити і встановити або відразу запустити, без установки. При запуску програма відображає вікно з ID і паролем для доступу до комп'ютера, а також TeamViewer дозволяє підключитися до іншого комп'ютера задавши його ID і пароль (рис. 1.19).

Таблиця 1.1

#### **Порівняльний аналіз програм віддаленого доступу**

<b>Характеристики</b>	<b>Назва програм віддаленого доступу</b>			
	<b>TeamViewer</b>	<b>LiteManager</b>	<b>Ammy admin</b>	<b>RAdmin</b>
Протокол	Власне ПЗ	Власне ПЗ, RDP	Власне ПЗ, RDP	Власне ПЗ
Режими роботи	Клієнт & Сервер	Клієнт & Сервер & Gateway	Клієнт & Сервер	Клієнт & Сервер &
Вбудоване шифрування	AES-256	AES-256, RSA-2048	AES-256, RSA	AES-256
Передача файлів	+	+	+	+
Передача звуку	+	+	+	+
Багатоклієнтний режим	+	+	+	+
Віддалений помічник	+	+	+	+

Характеристики	Назва програм віддаленого доступу			
	TeamViewer	LiteManager	Ammy admin	RAdmin
Запит прав доступу	+	+	+	+
Linux клієнт	+	-	-	+
Mac OS клієнт	+	+	-	-
MS Windows клієнт	+	+	+	+
Android клієнт	+	+	-	-
Фіксований ID	+	+	+	-

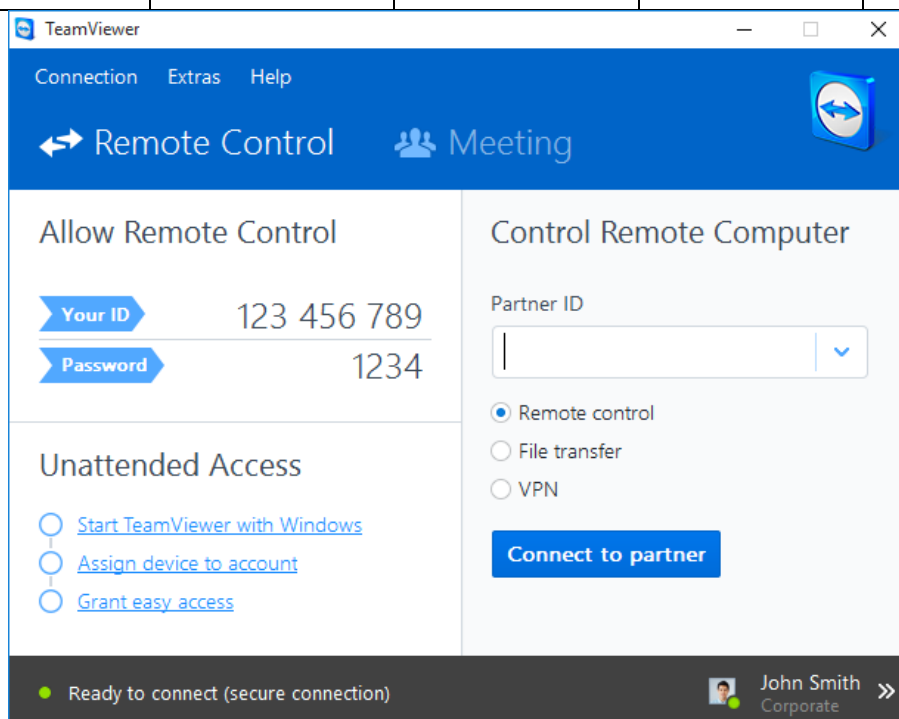


Рис. 1.19. Діалогове вікно програми TeamViewer

### *Переваги.*

Програма має ряд основних режимів роботи, таких як віддалене керування, передача файлів, чат та демонстрація робочого столу. Також програма дозволяє налаштувати цілодобовий доступ до комп'ютера, що буде зручно для системного адміністрування. Є версії для всіх мобільних платформ, для різних операційних систем. Простий і цілком зрозумілий інтерфейс плюс ряд додаткових утиліт для розширення функціоналу програми, будуть корисні для служб віддаленої підтримки.

### *Недоліки.*

Програма є безкоштовною тільки для некомерційного використання. При роботі з програмою більше 5 хвилин можуть виникнути труднощі, наприклад TV може заблокувати сеанс віддаленого підключення, розпізнавши його як комерційне використання. Для цілодобового віддаленого доступу або адміністрування декількох комп'ютерів, комп'ютерної мережі, доведеться платити за додаткові модулі програми. Вартість програми висока.

#### *Підсумок.*

Дана програма ідеально підходить для разового віддаленого підключення або використання її нетривалий період часу. Зручно використовувати з мобільних платформ, але не адмініструвати велику кількість комп'ютерів. За додаткові модулі доведеться доплачувати.

#### **LiteManager**

Проста, але досить таки потужна по можливостях програма, складається з двох частин, перша це Server який потрібно встановити або запустити на віддаленому комп'ютері і Viewer, який дозволяє керувати іншим комп'ютером. Для роботи програма вимагає трохи більше навичок і досвіду від користувача, хоча робота сервером навіть простіше ніж в TeamViewer, сервер можна один раз встановити і більше не яких дій від користувача не потрібно, ID буде завжди постійний, його навіть можна задати самому, що дуже зручно для запам'ятовування. Версія LiteManager Free є безкоштовною для особистого та комерційного використання (рис. 1.20).

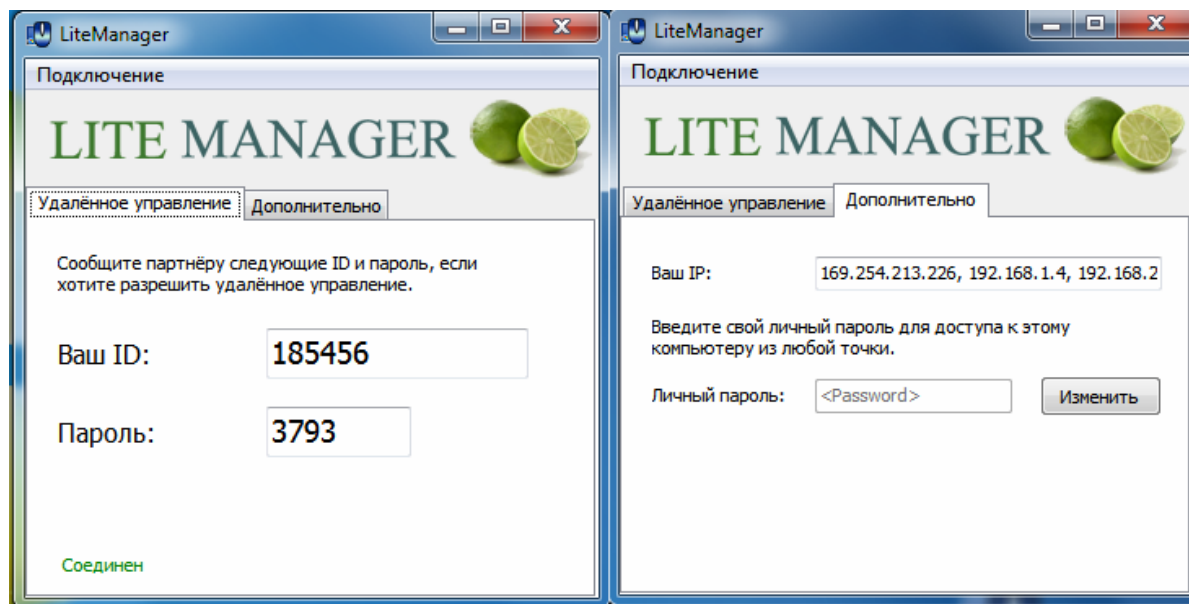


Рис. 1.20. Діалогове вікно програми LiteManager

### *Переваги.*

У програмі крім основних режимів віддаленого доступу (віддаленого управління, передачі файлів, чату, диспетчера задач, редактора реєстру) є специфічні функції, наприклад: інвентаризація, запис екрану, віддалена установка. Програма безкоштовна для використання на 30-ти комп'ютерах, її можна використовувати для цілодобового доступу без будь-яких додаткових модулів. Відсутні будь-які обмеження по часу роботи. Є можливість налаштування свого власного ID сервера для налаштування корпоративної служби підтримки. У програмі немає жодних обмежень за часом роботи і блокувань.

### *Недоліки.*

Відсутність клієнта для мобільних платформ та інші системи, обмеження на 30 комп'ютерів у безкоштовній версії, а також необхідність придбання ліцензії для адміністрування більшої кількості, є обмеженнями програми. Деякі специфічні режими роботи доступні лише в Pro версії.

### *Підсумок.*

Програма LiteManager підійде для надання віддаленої підтримки, для адміністрування декількох десятків комп'ютерів абсолютно безкоштовно, для настройки власної служби віддаленої підтримки. Вартість програми найнижча

в своєму сегменті і ліцензія не обмежена за часом.

### **Ammy admin**

Програма в основному аналогічна TeamViewer, але більш простий варіант. Присутні тільки основні режими роботи - перегляд і управління, передача файлів, чат. Програма може працювати без установки, безкоштовна для некомерційного використання (рис. 1.21).

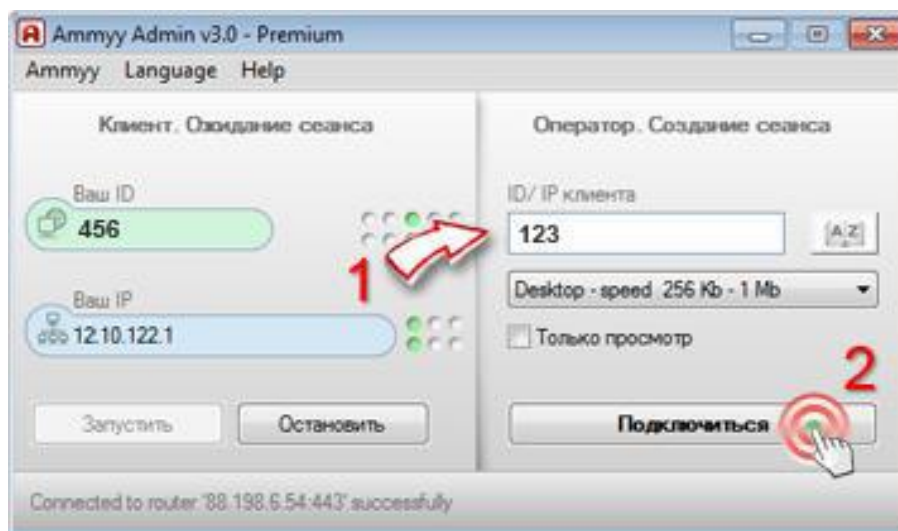


Рис. 1.21. Діалогове вікно програми Ammy admin

#### *Переваги.*

Проста і легка програма, можна працювати як в Інтернеті, так і в локальній мережі, потребує мінімальних налаштувань і не вимагає якихось особливих умінь і навичок. За порівняння з TeamViewer м'якша ліцензійна політика.

#### *Недоліки.*

Мінімум функцій для віддаленого управління, адмініструвати великий парк комп'ютерів буде складно. При довгому використанні, більше 15 годин на місяць, сеанс роботи може бути обмежений або заблокований, платна для комерційного використання.

#### *Підсумок:*

Дана програма більше підійде для разового підключення до комп'ютера і не сильно складних завдань, наприклад в якості надання допомоги не досвідченому користувачеві в налаштуванні комп'ютера.

### **RAdmin**

Одна з перших програм віддаленого управління і відома у своєму колі, більше призначена для системного адміністрування, основний акцент зроблено на безпеці. Програма складається з двох частин: компонент сервера і клієнта. Вимагає установки, не досвідченому користувачеві буде не просто з нею розібратися, програма призначена в основному для роботи з IP адресою, що не зовсім зручно для надання техпідтримки через Інтернет. Програма платна, але має безкоштовний тестовий період (рис. 1.22).

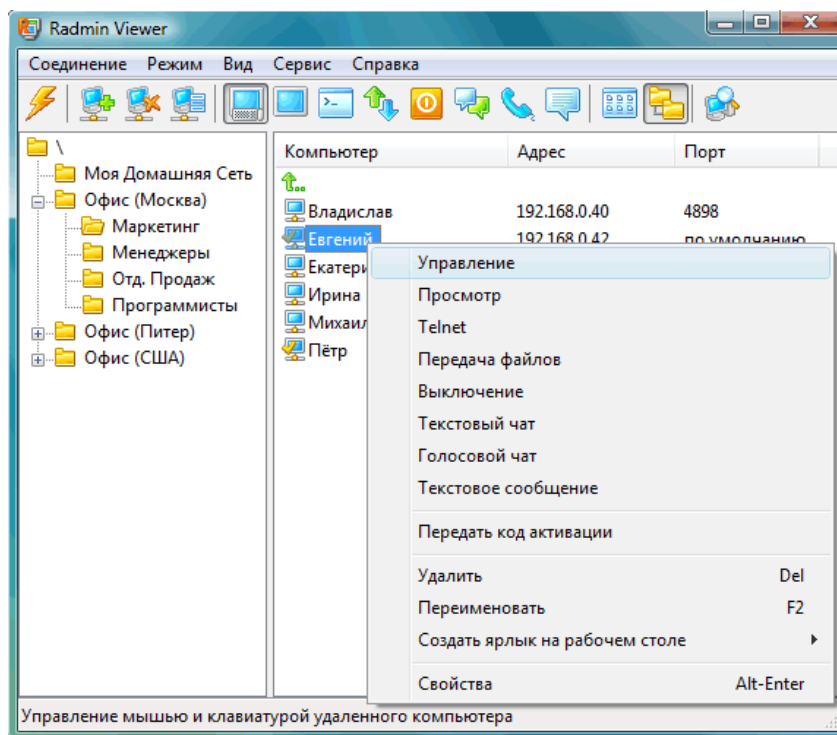


Рис. 1.22. Діалогове вікно програми RAdmin

### *Переваги.*

У програми висока швидкість роботи, особливо в хорошій мережі, завдяки відео драйверу захоплення робочого столу. Крім того, перевагою є підвищена надійність і безпека. Вбудована технологія Intel AMT надає можливість віддаленого підключення до BIOS комп'ютера та його налаштування. Вона реалізована лише з основними режимами роботи, такими як віддалене управління, передача файлів, чат і таке інше.

### *Недоліки.*

Практично неможливо працювати без призначеної IP-адреси, що означає обов'язок підключення через ідентифікатор (ID). Відсутній клієнт для

мобільних пристроїв, а використання програми можливе лише після тестового періоду тривалістю 30 днів, оскільки вільна версія відсутня. Робота з програмою передбачає наявність навичок у досвідченого користувача, а при підключенні відеодрайвер може вимагати відключення графічної оболонки Aero.

#### *Підсумок.*

Програма більше підійде для системних адміністраторів для адміністрування комп'ютерів і серверів в локальній мережі. Для роботи через Інтернет, можливо, доведеться налаштувати VPN тунель.

Таким чином, проведений аналіз довів, що сьогодні існує доволі велика кількість програм, що забезпечують віддалений доступ та віддалене адміністрування. Але, більшість з них потребує наявності (придбання) ліцензії для довготривалої роботи з програмою. Тому розробка власного додатку віддаленого доступу може вирішити цю проблему.

#### **Висновки до розділу**

В першому розділі магістерської роботи було досліджено теоретичні основи віддаленого доступу до ресурсів комп'ютера. Встановлено, що віддалений доступ являє собою функцію, що дозволяє користувачеві підключатися до комп'ютера через Інтернет за допомогою іншого ПК. Одним з видів віддаленого доступу є доступ до віддаленого робочого столу (Remote Desktop) - це термін, яким позначається режим управління, коли один комп'ютер отримує права адміністратора по відношенню до іншого, віддаленого. Зв'язок між пристроями відбувається в реальному часі за допомогою Інтернет або локальної мережі.

Доступ до віддаленого робочого столу здійснюється на базі архітектури «сервер - термінал», що дозволяє працювати з ресурсами сервера так само, як і з локальними ресурсами: виконувати команди, працювати з файловою системою, запускати додатки і т. д.

Проаналізовано принципи роботи протоколів віддаленого доступу: RDP, VNC, ICA. На підставі проведеного аналізу було вирішено обрати для

розробки проекту протоколи RDP і VNC, оскільки їх застосування дозволить реалізувати віддалений доступ не тільки до комп'ютерів з операційною системою Windows, а також дасть можливість застосовувати різні платформи на клієнті і сервері віддаленого доступу. Крім того, дані протоколи забезпечують високий рівень безпеки.

Проведений аналіз додатків (TeamViewer, LiteManager, Ammy admin, RAdmin) довів, що сьогодні існує доволі велика кількість програм, що забезпечують віддалений доступ та віддалене адміністрування. Але, більшість з них потребує наявності (придбання) ліцензії для довготривалої роботи з програмою. Тому розробка власного програмного додатку віддаленого керування робочою станцією може вирішити цю проблему.

## **РОЗДІЛ 2. РОЗРОБКА ПРОГРАМНОГО ДОДАТКУ ВІДДАЛЕНОГО КЕРУВАННЯ РОБОЧОЮ СТАНЦІЄЮ НА ПЛАТФОРМІ .NET**

### **2.1. Характеристика платформи .NET та обґрунтування інструментів розробки додатку**

Технологія .NET Framework є платформою, яка сприяє розробці та виконанню нового покоління програм. При розробці .NET Framework були враховані наступні цілі:

- забезпечити єдине середовище об'єктно-орієнтованого програмування для локального та розподіленого виконання об'єктного коду, яке може бути використане локально або в Інтернеті, або навіть віддалено;
- створити середовище виконання коду, яке мінімізує конфлікти при розгортанні програмного забезпечення та управлінні версіями;
- розробити середовище виконання коду, що гарантує безпечне виконання коду, навіть якщо він створений стороннім виконавцем, якому можна не повністю довіряти;
- створити середовище виконання коду, яке уникне проблем продуктивності середовищ виконання сценаріїв або інтерпретованого коду;
- забезпечити єдино-образні принципи розробки для різних типів додатків, таких як додатки для Windows і веб-додатки;
- забезпечити взаємодію на основі промислових стандартів, щоб гарантувати інтеграцію коду платформи .NET Framework з будь-яким іншим кодом.

*Архітектура .NET Framework* включає дві основні частини. Перша - це виконавче середовище Common Language Runtime (CLR), яке може виконувати звичайні та серверні програми. Серед CLR управляє пам'яттю, виконанням потоків, виконанням коду, перевіркою безпеки коду, компіляцією і іншими системними службами. Ці засоби є внутрішніми для керованого коду, який виконується в середовищі CLR.

Іншою ключовою складовою є бібліотека класів Framework Class Library (FCL), що включає безліч компонентів для оптимізованої роботи з базами даних, мережею, введенням/виведенням, файлами, інтерфейсами для користувача і багато іншого. Це дозволяє розробникам уникати низькорівневого програмування, скориставшись готовими класами.

Важливі аспекти бібліотеки класів включають:

- Windows Forms: відповідає за розробку графічного інтерфейсу і є обгорткою над Win32 API.
- ADO.NET: надає доступ до даних і в основному використовується для роботи з базами даних.
- ASP.NET: технологія розробки веб-сайтів, веб-додатків і веб-сервісів.
- Language Integrated Query (LINQ): реалізація мови запитів, що нагадує SQL, для програм на .NET.
- Windows Presentation Foundation (WPF): система створення графічних інтерфейсів, використовуючи мову розмітки XAML і графічну технологію DirectX.
- Windows Communication Foundation (WCF): система обміну даними між додатками .NET для створення розподілених додатків.

ADO.NET включає набір класів, які надають програмні інтерфейси для зручного підключення до баз даних незалежно від конкретної системи управління базами даних, їх структури та місця розташування. Технологія ADO.NET також дозволяє працювати з базою даних в режимі "розриву" з'єднання, що забезпечує ефективне використання ресурсів.

В об'єктній моделі ADO.NET можна виділити декілька рівнів (рис. 2.1).

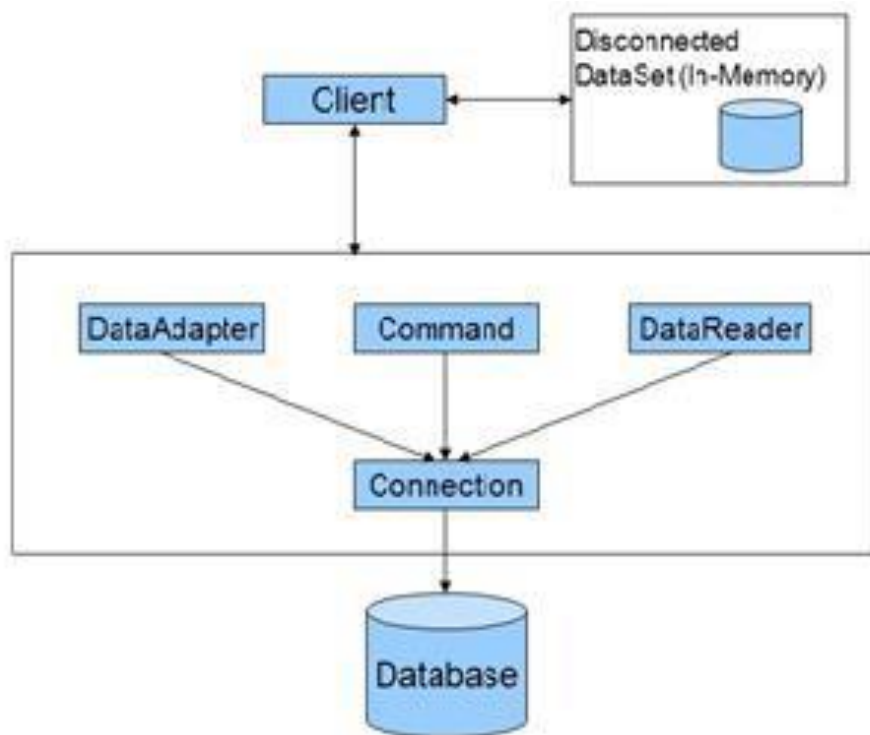


Рис. 2.1. Модель технології ADO.NET

*Рівень даних.* Це базовий рівень, на якому розташовуються самі дані. На цьому етапі забезпечується фізичне зберігання інформації на магнітних носіях, а також обробка даних на рівні вихідних таблиць, включаючи вибірку, сортування, додавання, видалення та оновлення.

*Рівень бізнес-логіки* представляє собою набір об'єктів, які визначають, яку базу даних слід використовувати для встановлення зв'язку і які конкретні дії необхідно виконати з даними, що містяться в цій базі. Для встановлення зв'язку з базами даних використовується об'єкт `FbDataConnection`. Для зберігання команд, що виконують будь-які дії над даними, використовується об'єкт `FbDataAdapter`. Для зберігання результатів вибірки використовується об'єкт `DataSet`.

*Рівень додатку* складається з набору об'єктів, які забезпечують можливість зберігати і відображати дані на комп'ютері кінцевого користувача. Для зберігання інформації використовується об'єкт `DataSet`, а для відображення даних використовується різноманітний набір елементів управління, таких як `DataGrid`, `TextBox`, `ComboBox`, `Label` та інші.

*Мови програмування та платформа .NET.* Однією з ключових

концепцій Microsoft .NET є можливість взаємодії програмних компонент, що написані різними мовами програмування. Наприклад, сервіс, розроблений на C++, може викликати метод класу із бібліотеки, написаної на Delphi; клас може бути написаний на C#, успадковано від класу, створеного на Visual Basic .NET, і виняток, який породжується методом у C#, може бути перехоплено та оброблено в Delphi. Кожна бібліотека (.NET assembly) має інформацію про свою версію, що дозволяє уникнути можливих конфліктів між різними версіями бібліотек.

Розглянемо основні мови програмування, які постачаються разом з Microsoft Visual Studio:

- C#
- Visual Basic .NET
- JScript .NET
- C++/CLI (нова версія Managed C++)
- F# (частина сімейства мов програмування ML, включена в VS2010/VS2012/VS2015/VS2017)

C# є найбільш популярною мовою програмування на платформі .NET. Це об'єктно-орієнтована мова зі строгою типізацією, яка дозволяє розробникам створювати безпечні і надійні додатки для платформи .NET Framework. C# використовується для створення різноманітних додатків, таких як клієнтські програми для Windows, XML-веб-служби, розподілені компоненти, клієнт-серверні додатки, додатки баз даних і багато інших. Visual C# надає розширений редактор коду, зручні конструктори інтерфейсу користувача, вбудований відладчик та інші інструменти, що полегшують процес розробки додатків на платформі .NET Framework.

Програми C # виконуються на платформі .NET Framework, яка інтегрована в Windows і містить віртуальне загальномовне середовище виконання (середу CLR) і уніфікований набір бібліотек класів.

Вихідний код, написаний на мові C#, проходить компіляцію в проміжну мову (IL), яка відповідає стандартам Common Language Infrastructure (CLI).

Код на мові ІЛ, а також ресурси, такі як точкові малюнки і рядки, зберігаються на диску у вигляді виконуваного файлу, який зазвичай має розширення .exe або .dll. Цей файл відомий як збірка і включає в себе маніфест із відомостями про типи, версії, вимоги безпеки, мову та регіональні параметри для цієї збірки.

Під час виконання програми на С#, Common Language Runtime (CLR) завантажує збірку і виконує різні операції відповідно до вмісту маніфесту. Якщо всі вимоги безпеки виконані, CLR виконує Just-In-Time (JIT) компіляцію з коду мови ІЛ в інструкції машинного коду. Крім того, CLR виконує інші завдання, такі як автоматичне прибирання сміття, обробка винятків і управління ресурсами. Термін "керований код" використовується для позначення коду, який виконується середовищем CLR, щоб відзначити його відмінність від "некерованого коду", який компілюється безпосередньо в машинний код для конкретної системи.

На схемі (рис. 2.2) відображено зв'язок між файлами вихідного коду С#, бібліотеками класів .NET Framework, збірками і середовищем CLR.

Таким чином, застосування мови програмування С# дозволить найбільш ефективно реалізувати можливості .NET Framework для розробку додатку віддаленого доступу.

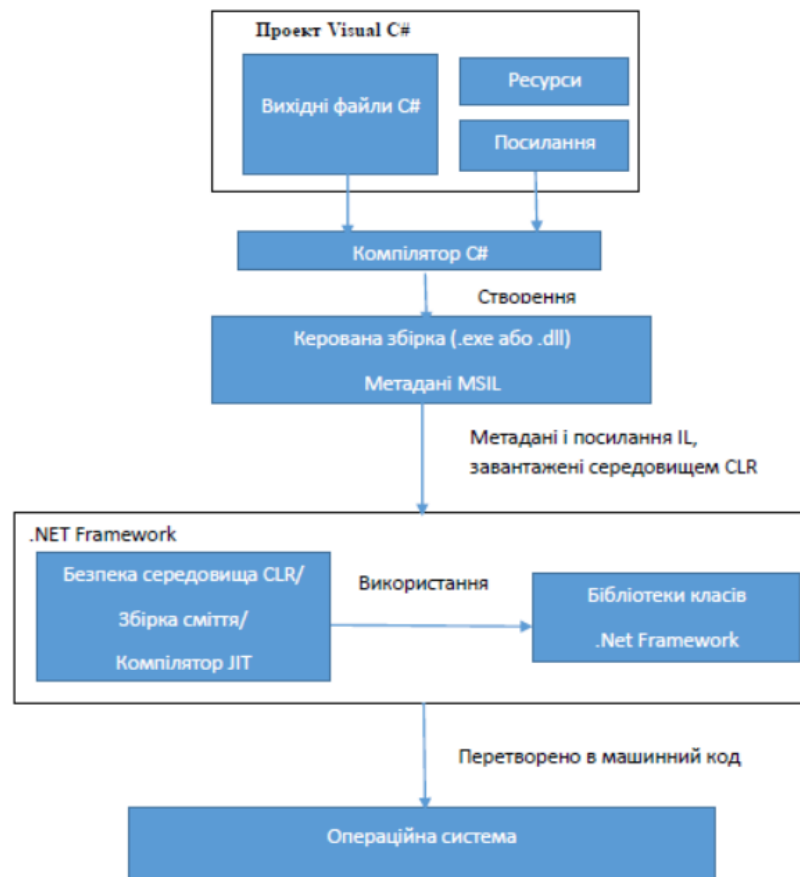


Рис. 2.2. Зв'язок між додатком C # та платформою .NET Framework

## 2.2. Розробка програмного додатку

### Постановка задачі

Після аналізу всіх вимог були сформульовані цілі:

1. Розробити клієнт-додаток, що дозволяє:
  - здійснювати віддалений доступ за протоколами RDP і VNC.
  - підключатися до хосту, дані про який зберігаються в файлі (\* .rdp).
  - спостерігати і керувати віддаленим комп'ютером в мережі.
  - зберігати дані про вузол підключення в базі даних.

Клієнт повинен бути побудований так, щоб реалізовувати варіанти використання, зображені на рис. 2.3

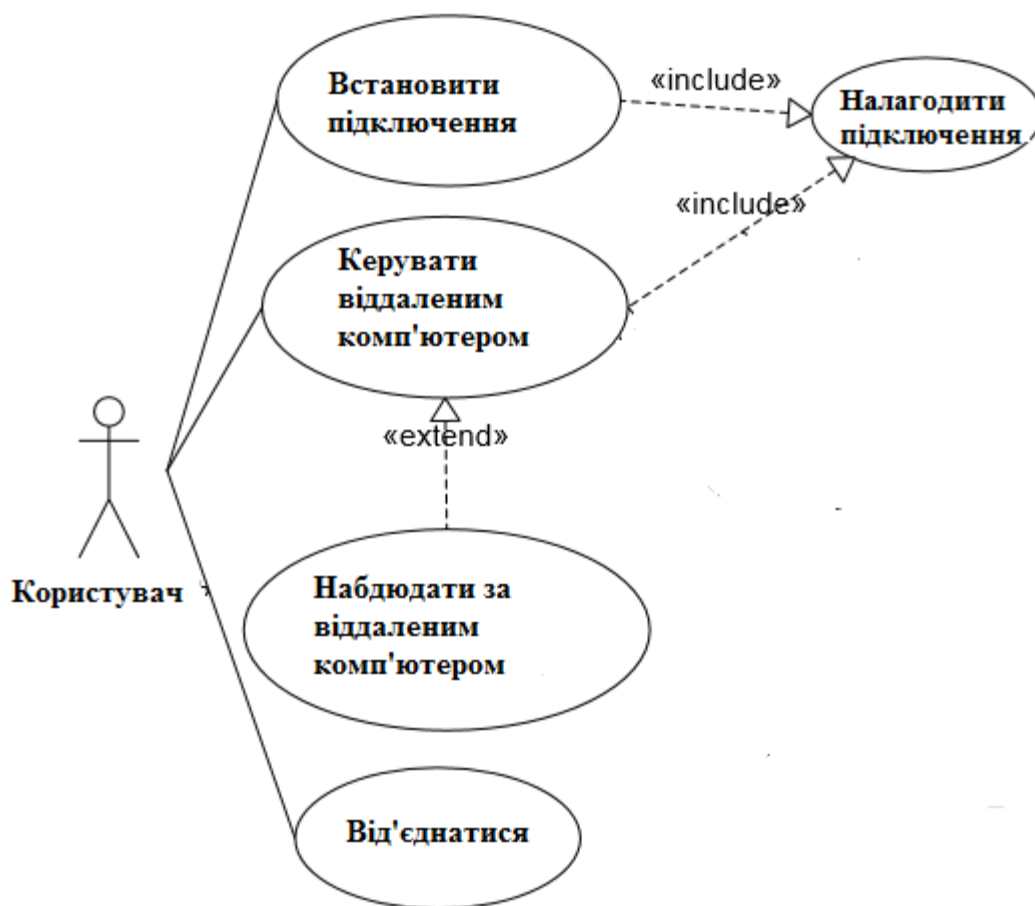


Рис. 2.3. Функціональна схема програмного додатку віддаленого керування робочою станцією

Під час розробки використовувати наступні засоби програмування:

- мова програмування C # і середовище розробки Microsoft Visual Studio 2010;
- для реалізації інтерфейсної частини додатку застосовується бібліотека Windows Forms на основі технології .Net Framework 4.
- в реалізації моделі доступу до даних застосовується технологія ADO.NET.

### Архітектура проєкту

Для реалізації клієнта було вирішено взяти за основу трирівневу архітектуру, кожен шар якої буде відповідати за свою логічну частину:

- Рівень відображення
- Рівень логіки - управління підключенням
- Рівень протоколу

Рівень відображення відповідає тільки за візуалізацію поточного екрану віддаленого комп'ютера і дозволяє користувачеві здійснювати дії миші і клавіатури, які будуть передані на нижній рівень.

Завдання рівня логіки полягає в управлінні всім процесом роботи програми в цілому і кожного його компонента. В першу чергу мова йде про встановлення підключення до віддаленого комп'ютеру.

На самому нижньому рівні відбувається логіка дії протоколів. Концептуальна модель класів клієнтського додатку відображена у UML-діаграмі (рис. 2.4). Основні класи проекту - клас RemoteRDPClient та клас RemoteVNCClient беруть участь в реалізованій ієрархії класів і є нащадками абстрактного класу RemoteAbstractWidget.

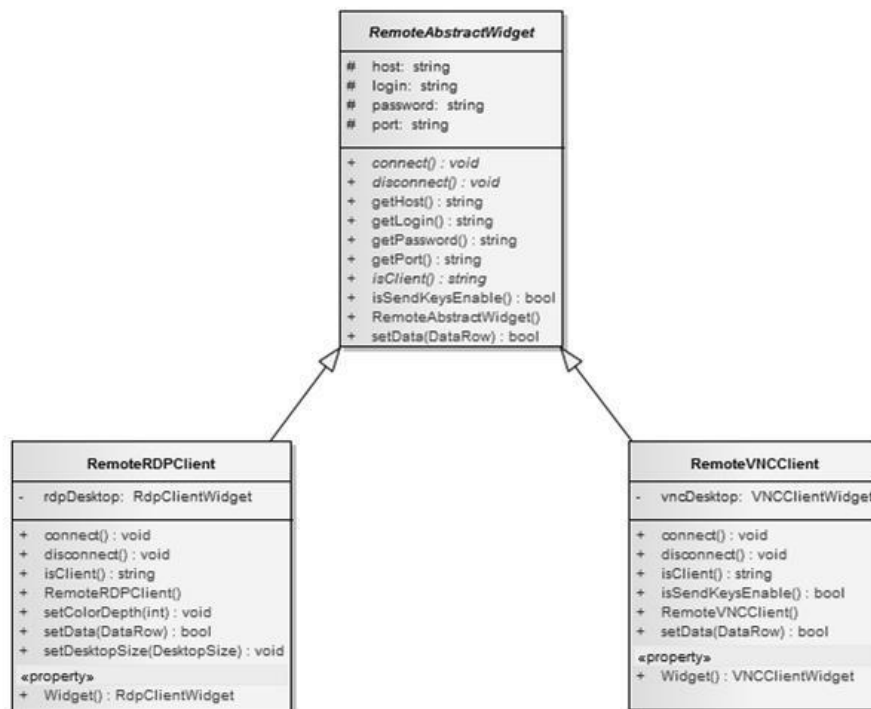


Рис. 2.4. Ієрархічна модель класів

Клас RemoteAbstractWidget є абстракцією поведінки класів RemoteRDPClient та RemoteVNCClient. Код класу RemoteAbstractWidget наведено нижче:

```

public abstract class RemoteAbstractWidget
{
    protected string

```

Лістинг 2. 1

```

password; protected
string login; protected
string host; protected
string port;

    public
RemoteAbstractWidget() { }
public abstract void connect();
public abstract void disconnect();
public abstract string isClient();
        public virtual bool setData(DataRow row)
        {
45, true);
        host = Convert.ToString(row["ip"]); port =
Convert.ToString(row["port"]); login =
Convert.ToString(row["login"]);
        password = Helper.XorString(Convert.ToString(row["password"]),
return true;
    }

        public virtual bool isSendKeysEnable()
        {
            return false;
        } public string getPassword()
        {
            return password;
        }
        public string getLogin()
        {
            return login;
        }
        public string getPort()

```

```

    {
        return port;
    }

    public string getHost()
    {
        return host;
    }
}

```

### Реалізація інтерфейсу додатка

Інтерфейсна частина додатку повинна складатися з форми, яка містить наступні основні елементи управління:

- Рядок меню.
- Панель інструментів.
- Список.
- Картотека.
- Рядок стану.

Після всіх налаштувань розміщених компонентів на головній формі форма прийняла вид, представлений на рис. 2.5.

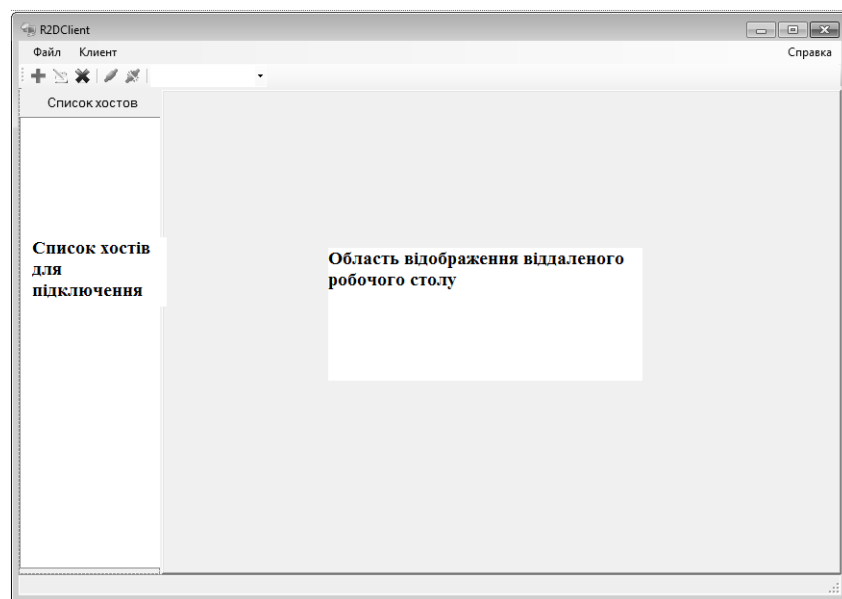


Рис. 2.5. Головна форма проєкту

### Реалізація рівня відображення

У центральній області форми при ініціалізації підключення і в залежності від обраного типу підключення програмними засобами було реалізовано розміщення на формі наступних компонентів:

- AxMSTSCLib.AxMsRdpClientNotSafeForScripting
- VncSharp.RemoteDesktop

Компонент AxMSTSCLib.AxMsRdpClientNotSafeForScripting та VncSharp.RemoteDesktop відповідають за відображення віддаленого робочого стола відповідно до типу підключення RDP або VNC

Ієрархія класів, зазначена на рис. 2.4, реалізована для уніфікації компонентів AxMSTSCLib.AxMsRdpClientNotSafeForScripting і VncSharp.RemoteDesktop, тому що в цих компонентах визначені різні, не подібні між собою методи підключення. Це дозволило використовувати поліморфні властивості в подальшій програмній реалізації.

Приклад коду, що реалізує відображення віддаленого столу при підключенні за протоколом VNC наведено нижче:

Лістинг 2.2

```
using VNCClientWidget = VncSharp.RemoteDesktop;

public RemoteVNCClient() : base()
{
    vncDesktop = new VNCClientWidget();
    vncDesktop.Enabled = true;
    vncDesktop.Visible = true;
    vncDesktop.AutoScroll = true;
    vncDesktop.Dock = System.Windows.Forms.DockStyle.Fill;
    vncDesktop.Location = new System.Drawing.Point(0, 0);
    vncDesktop.Name = "vncDesktop";
    vncDesktop.TabIndex = 0;
    vncDesktop.GetPassword = base.getPassword;
    vncDesktop.SetScalingMode(true);
```

```
}
```

В цьому методі створюється об'єкт `vncDesktop` класу `ncSharp.RemoteDesktop`: Код для RDP протоколу має аналогічну структуру.

### **Реалізація рівня логіки**

#### *Реалізація технології віддаленого доступу*

Нижче представлений лістинг коду методу класу-обробнику подій, який викликається при підключенні, використовуючи RDP протокол.

Лістинг 2.3

```
private void rdpConnectMenuItem_Click (object sender, EventArgs e)
{
    RemoteRDPCClient client = new RemoteRDPCClient();
   TabPage NewPage = new
    TabPage(); NewPage.Text =
    Convert.ToString(ipList.Items[ipList.SelectedIndex]
    ); NewPage.Controls.Add(client.Widget);
    tabControl.TabPages.Add(NewPage);
    tabControl.SelectedIndex = tabControl.TabCount -
    1; DataTable table = SQL.dataTable("hostdata");
    DataRow row = table.Rows[ipList.SelectedIndex];
    if (client.setData(row))
    {
        client.setColorDepth(RDPSettingsDlg.getColorDepth());
        client.setDesktopSize(RDPSettingsDlg.getDesktopSize()
        ); client.connect();
    }
    else disconnectButton_Click(sender, e);
}
```

У наведеному лістингу програмної реалізації методу створюється об'єкт класу `RemoteRDPCClient`. Далі об'єкту `NewPage.Text` присвоюється значення адреси хоста, після чого викликом функції `NewPage.Controls.Add`

(client.Widget) вміст вкладки заповнюється об'єктом класу RdpClientWidget. Властивість client.Widget повертає цей об'єкт. Клас RdpClientWidget є нащадком класу AxMsRdpClientNotSafeForScripting. Далі створена вкладка додається на панель вкладок, виконавши наступну інструкцію tabControl.TabPages.Add (NewPage). Після чого з бази даних отримуємо налаштування підключення (адреса хоста, ім'я користувача, пароль) і проводиться підключення з урахуванням налаштувань екрану викликом методу client.connect (). Метод connect() по протоколу RDP наведено нижче.

Лістинг 2.4

```
public override void connect()
{
    if (host != null)
    {
        try
        {
            vncDesktop.VncPort = Convert.ToInt32(base.getPort());
            vncDesktop.Connect(host, false, true);
            if (vncDesktop.IsConnected)
            {
                vncDesktop.SetScalingMode(true);
                vncDesktop.FullScreenUpdate();
            }
            else
            {
                MessageBox.Show("Не удается осуществить соединение.
                Проверьте правильность ввода пароля", "Ошибка соединения",
                MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
        }
        catch (VncProtocolException vex)
        {
            MessageBox.Show(string.Format("Unable to connect to VNC
```

```

host:\n\n{0}.\n\nCheck that a VNC host is running there.", vex.Message),
        string.Format("Unable to Connect to {0}", host),
        MessageBoxButtons.OK,
        MessageBoxIcon.Exclamation);
    }
    catch (Exception ex)
    {
        MessageBox.Show(string.Format("Unable to connect to
host. Error was: {0}", ex.Message),
        string.Format("Unable to Connect to {0}", host),
        MessageBoxButtons.OK,
        MessageBoxIcon.Exclamation);
    }
}
}
}

```

Наведений у лістингу 2.3 та 2.4 код дозволяє здійснити з'єднання з віддаленим хостом, використовуючи протокол RDP.

Аналогічним чином реалізовано підключення з використанням протоколу VNC. Виклик метода connect() (лістинг 2.5) відбувається у обробнику подій vncConnectMenuItem\_Click(object sender, EventArgs e).

Лістинг 2.5

```

public override void connect()
{
    if (host != null)
    {
        try
        {
            vncDesktop.VncPort = Convert.ToInt32(base.getPort());
            vncDesktop.Connect(host, false, true);
            if (vncDesktop.IsConnected)

```

```

        {
            vncDesktop.SetScalingMode(true);
            vncDesktop.FullScreenUpdate();
        }
    Else
        MessageBox.Show("Не удастся осуществить соединение.
        Проверьте правильность ввода пароля", "Ошибка соединения",
        MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
    }
    catch (VncProtocolException vex)
    {
        MessageBox.Show(string.Format("Unable to connect to VNC
        host:\n\n{0}.\n\nCheck that a VNC host is running there.", vex.Message),
            string.Format("Unable to Connect to {0}", host),
            MessageBoxButtons.OK,
            MessageBoxIcon.Exclamation);
    }
    catch (Exception ex)
    {
        MessageBox.Show(string.Format("Unable to connect to host.
        Error was: {0}", ex.Message),
            string.Format("Unable to Connect to {0}", host),
            MessageBoxButtons.OK,
            MessageBoxIcon.Exclamation);
    }
}
}
}

```

*Збереження даних про хости, що під'єднуються*

Для реалізації можливості зберігання даних про хости, що підключаються, була використана база даних SQLite і технологія доступу до

даних ADO.NET [1, с. 10].

Для створення бази даних використовується SQLite – компактна вбудована система управління базами даних (СУБД). SQLite відрізняється від парадигми клієнт-сервер, оскільки його движок не є окремим самостійно працюючим процесом, з яким програма взаємодіє. Замість цього, движок SQLite функціонує як бібліотека, з якою програма компілюється, і стає невід'ємною частиною самої програми. Таким чином, для обміну інформацією використовуються виклики функцій (API) бібліотеки SQLite. Такий підхід дозволяє зменшити накладні витрати, покращити час відгуку і спростити структуру програми. SQLite зберігає всю базу даних, включаючи її структуру, таблиці, індекси і дані, у єдиному стандартному файлі на комп'ютері, де виконується програма.

Для збереження даних про хости, що підключаються, створена таблиця hosts.db. Модель бази даних наведена у табл. 2.1

Таблиця 2.1

Модель бази даних hosts.db

Назва поля	Ім'я поля	Тип даних	Властивості
Унікальний ідентифікатор	ID	INTEGER	PRIMARY KEY, UNIQUE, NOT NULL,
Адреса IP	IP	VARCHAR(64)	UNIQUE, NOT NULL,
Назва порту	PORT	VARCHAR(16)	
Логін	LOGIN	VARCHAR(64)	
Пароль	PASSWORD	VARCHAR(64)	

Для взаємодії з об'єктами ADO.NET був розроблений клас SQLiteInterface, даний клас дозволяє абстрагуватися від інтерфейсів ADO.NET і надає простий функціонал доступу і оперування з даними бази даних. Нижче на лістингу 2.6 наведена реалізація одного з методів даного класу.

Лістинг 2.6.

```
private bool openDataBase()
{
    try { if (connection.ConnectionString.Length > 0)
        connection.Open();
```

```

        else throw new System.Exception("Строка соединения с
        БД не корректна."); }
    catch (System.Exception Except) {
        MessageBox.Show(Except.Message, "Ошибка");
        connection.Close();
        return false;
    }
    return connection.State == System.Data.ConnectionState.Open
        ? true : false;
}

```

Вище наведений метод здійснює підключення до бази даних, попередньо перевіряючи рядок з'єднання з базою даних.

При запуску програми здійснюється заповнення списку даними хостів з бази даних за допомогою об'єкта класу `SQLiteInterface` в конструкторі класу головної форми. Нижче на лістингу 2.7 представлений даний фрагмент коду програми.

Лістинг 2.7.

```

public MainWindow()
{
    SQL.connectionString = "Data Source=hosts.db; Version=3;";
    SQL.addTable("hostdata");
    SQL.fill("hostdata");
    DataTable table = SQL.dataTable("hostdata");
    for (int i = 0; i < table.Rows.Count; ++i)
        ipList.Items.Add(new IListBoxItem(table.Rows[i][1]
        .ToString(), 0));
    ipList.SelectedIndex = 0;
}

```

*Підключатися до хосту, дані про який зберігаються в файлі (\* .rdp).*

RDP-файл - це текстовий файл, що містить настройки підключення до

заданого сервера. Для реалізації можливості імпорту даних з \* .rdp файлів реалізований метод openMenuItem\_Click ():

Лістинг 2.8.

```
private void openMenuItem_Click(object sender, EventArgs e)
{
    OpenFileDialog myOpenFileDialog;
    myOpenFileDialog = new OpenFileDialog();
    myOpenFileDialog.Filter = "RDP File|*.rdp";
    myOpenFileDialog.Title = "Выбор файла с настройками RDP";
    myOpenFileDialog.ShowDialog();
    if (myOpenFileDialog.FileNames.Count() > 0)
    {
        RDPFile MyRdpFile = new RDPFile();
        MyRdpFile.Read(myOpenFileDialog.FileName)
        ; if (MyRdpFile.FullAddress != string.Empty)
        {
            DataRow row =
            SQL.newRow("hostdata"); row["ip"] =
            MyRdpFile.FullAddress; row["port"] =
            string.Empty;
            row["login"] = MyRdpFile.Username;
            row["password"] =
            Helper.XorString(MyRdpFile.Password, 45, true);
            SQL.addNewRow("hostdata", row);
            ipList.Items.Add(new IListBoxItem(MyRdpFile.FullAddress, 0));
        }
    }
}
```

У вище наведеному лістингу створюється об'єкт класу OpenFileDialog, який виводить на екран стандартний діалог відкриття файлу. Далі шлях до

вибраного файлу передається методу Read (myOpenFileDialog.FileName) викликається об'єктом класу RDPFile, здійснюється зчитування даних з файлу, після чого даний запис заноситься в базу даних і відображається в списку хостів на формі. Нижче на рис. 2.6 представлений діалог відкриття rdp файлу реалізований в програмі.

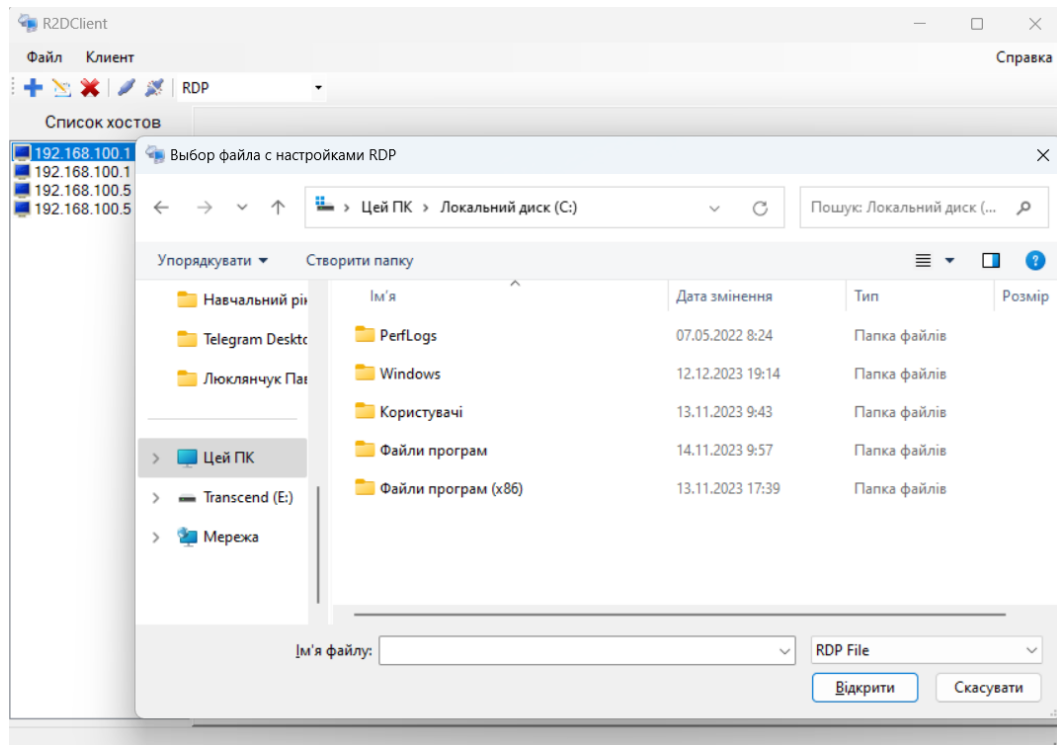


Рис. 2.6. Діалогове вікно вибору файлу

### Реалізація рівня протоколу

Реалізація цього рівня в програмі відбувається за допомогою застосування бібліотек Interop.MSTSCLib.dll і AxInterop.MSTSCLib.dll - модулі, що належить збірці бібліотек типів `MSTSCLib` для операційної системи Windows та VncSharp.dll – це OpenSource компонент, реалізація протоколу VNC для .NET Framework.

Таким чином, розроблений клієнтський додаток віддаленого доступу R2DClient повністю відповідає всім вимогам, що вказані у технічному завданні.

### 2.3. Тестування програмного додатку

Тестування розробленого додатку будемо здійснювати за допомогою встановленої віртуальної машини VirtualBox версія 5.2.22 [12]. Параметри

віртуальної машини та операційної системи подані нижче:

- операційна система - Windows XP (32-bit);
- тип підключення - віртуальний адаптер хоста;
- IP-адреса 192.168.56.2;
- маска підмережі 255.255.255.0;
- основний шлюз 192.168.56.1;
- у вікні «Властивості системи», вкладка «Віддалені сеанси» встановити опцію «Дозволити віддалений доступ цьому комп'ютеру».

На віртуальну машину становлено VNC-сервер TightVNC. Сервер налагоджено з паролем root.

Параметри комп'ютеру, на якій встановлено клієнтський додаток віддаленого доступу R2DClient:

- операційна система Windows 8.1 (64-bit);
- підключення - VirtualBox Host-Only Ethernet Adapter;
- IP-адреса 192.168.56.1;
- маска підмережі 255.255.255.0.

### **Цілі та методи тестування**

Клієнтський додаток віддаленого доступу R2DClient повинен забезпечувати:

1. Зберігання даних про вузол підключення в базі даних.
2. Підключення до віддаленого комп'ютеру за RDP протоколом та адміністрування.
3. Підключення до віддаленого комп'ютеру за VNC протоколом та адміністрування.
4. Підключення до хосту, дані про який зберігаються в файлі (\*.rdp).

В ході тестування будемо використовувати ручний метод. Вручну виконуване тестування (manual testing) представляє собою складову процесу контролю якості під час розробки програмного забезпечення, при якій використовується моделювання потенційних сценаріїв користувацької

взаємодії.

## Етапи тестування

### 1. Зберігання даних про вузол підключення в базі даних.

На клієнтському додатку встановимо тип протоколу - RDP, IP-адресу хосту, порт, логін та пароль доступу до комп'ютеру – admin, admin. Після виконання дій дані про хост збережено в БД та відображено у лівій панелі (рис. 2.7).

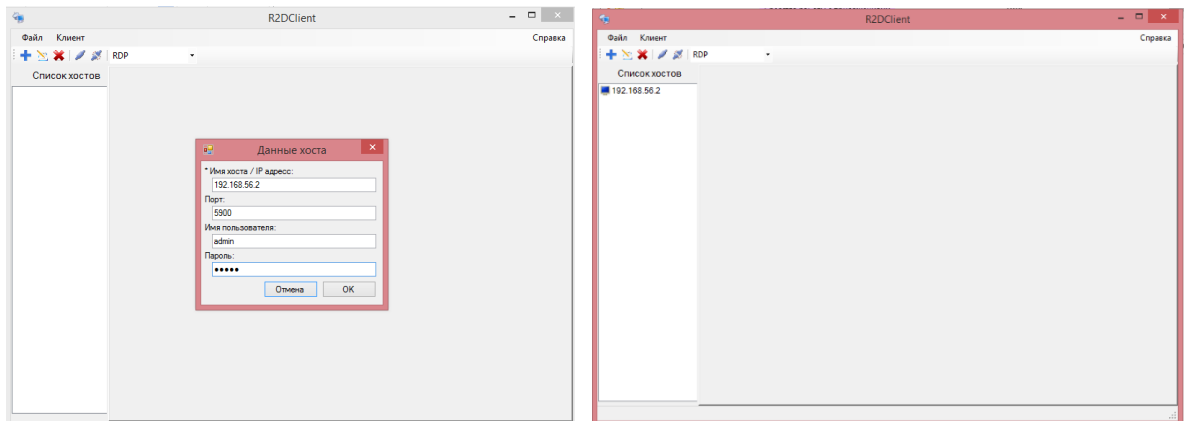


Рис. 2.7. Зберігання даних про вузол підключення в базі даних

### 2. Підключення до віддаленого комп'ютеру за RDP протоколом та адміністрування.

Тестування передбачає, що віртуальна машина налагоджена то запущена. Клієнтський додаток віддаленого доступу R2DClient запущено на машині клієнта. На клієнтському додатку встановимо тип протоколу - RDP, IP-адресу хосту, порт, логін та пароль доступу до комп'ютеру – admin, admin (рис. 2.8).

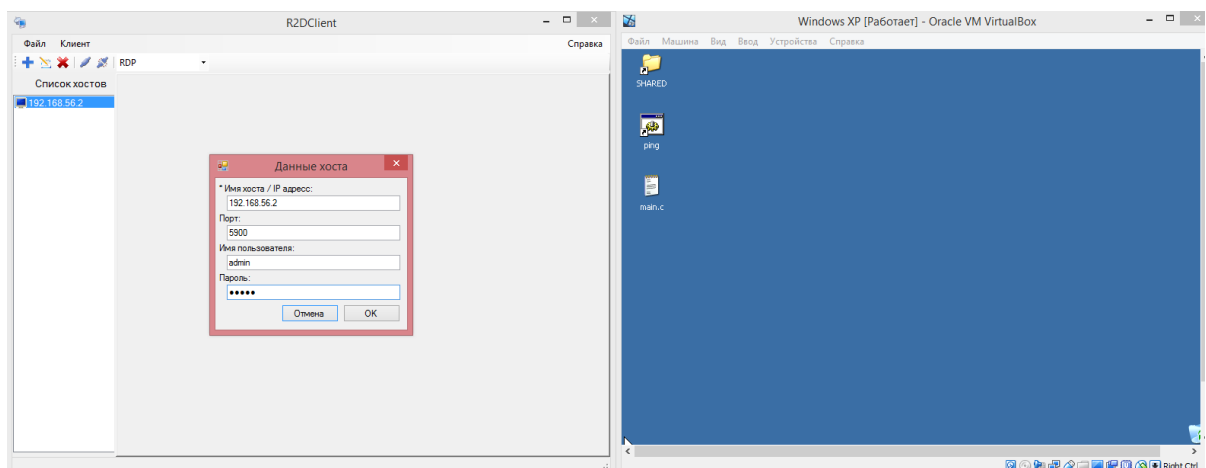


Рис. 2.8. Встановлення підключення за RDP протоколом

Після виконання операції з'єднання з хостом ми отримуємо відображення віддаленого робочого столу на клієнтському додатку (рис. 2.9)

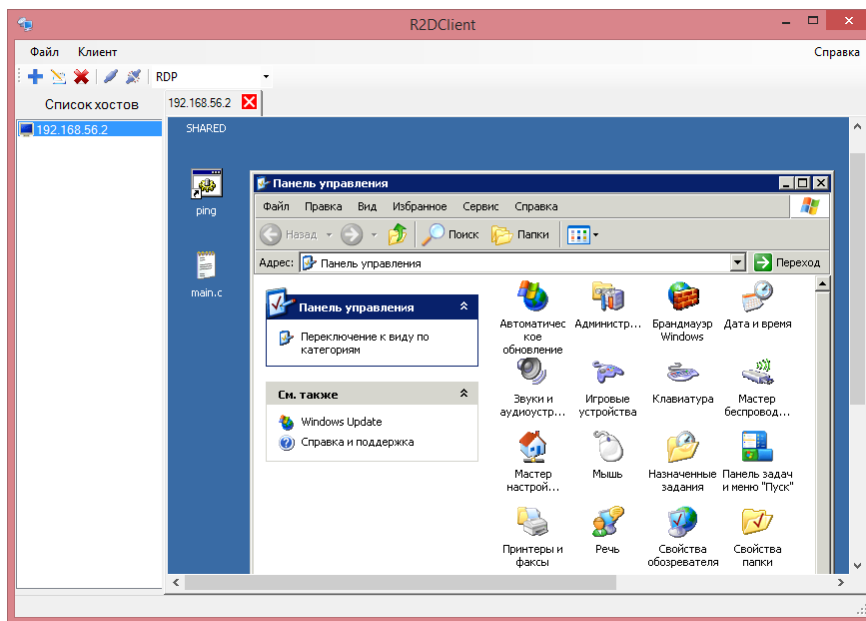


Рис. 2.9. Відображення віддаленого робочого столу на клієнтському додатку

Для тестування функцій адміністрування додамо нового користувача за допомогою опції Адміністрування – Керування комп'ютером в панелі інструментів на віддаленому комп'ютері. Початковий стан вкладки Локальні користувачі зображено на рис. 2.10.

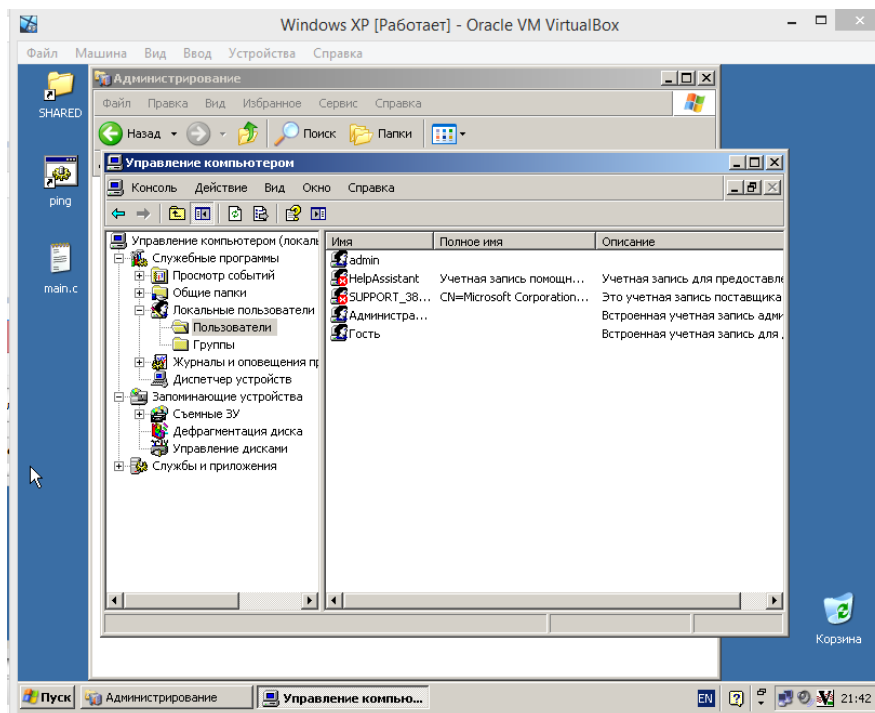


Рис. 2.10. Початковий стан вкладки Локальні користувачі

Виконаємо дії додавання користувача на клієнтському додатку (рис.

2.11).

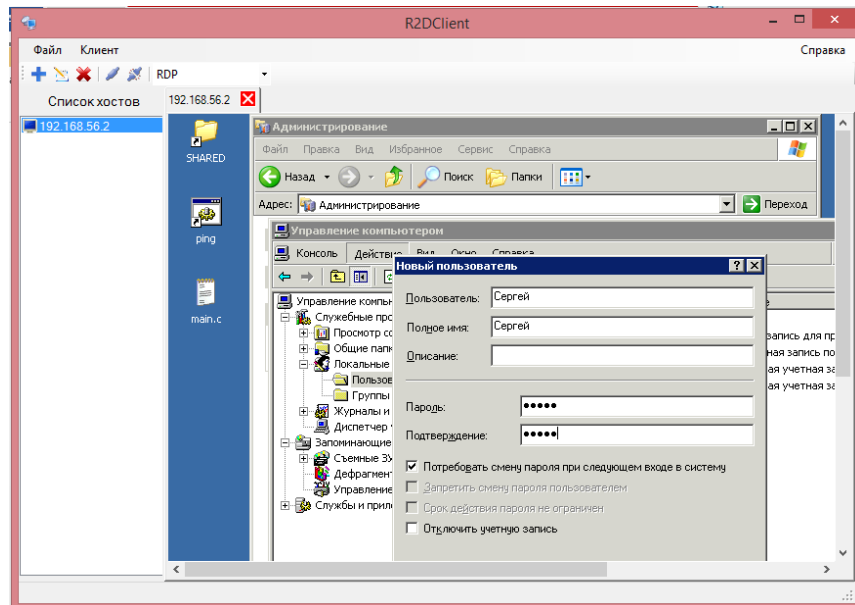


Рис. 2.11. Додавання користувача на клієнтському додатку

Результати віддаленого адміністрування показано на рис. 2.12.

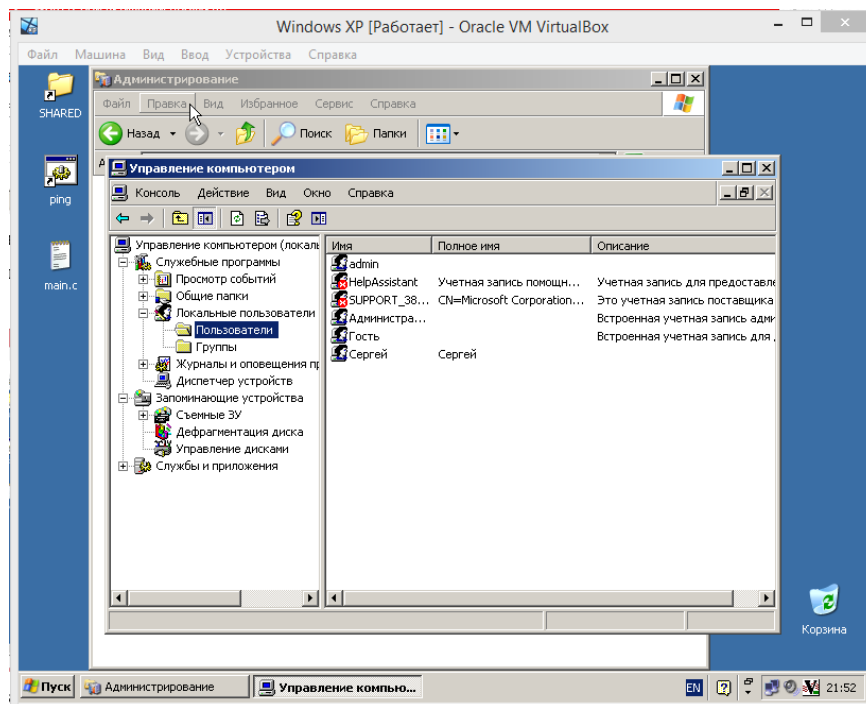


Рис. 2.12. Результати віддаленого адміністрування

3. Підключення до віддаленого комп'ютеру за VNC протоколом та адміністрування.

Запустимо VNC-сервер TightVNC на виконання. Пароль – root. На клієнтському додатку встановимо тип протоколу - VNC, IP-адресу хосту, порт, логін та пароль доступу до серверу (рис. 2.13).

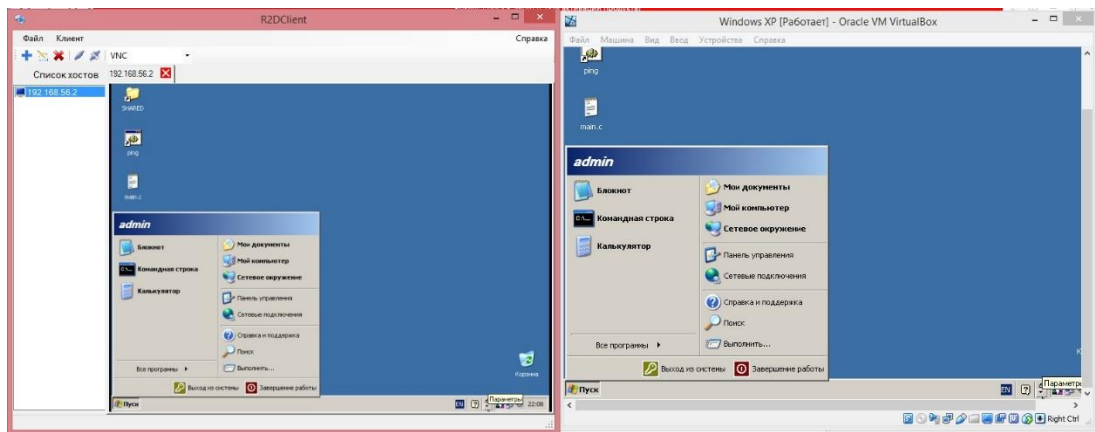


Рис. 2.13. Віддалений доступ до робочого столу за VNC протоколом

Для тестування функцій адміністрування видалимо нового користувача з іменем Сергій за допомогою Облікових записів користувачів панелі інструментів (рис. 2.14-2.15).

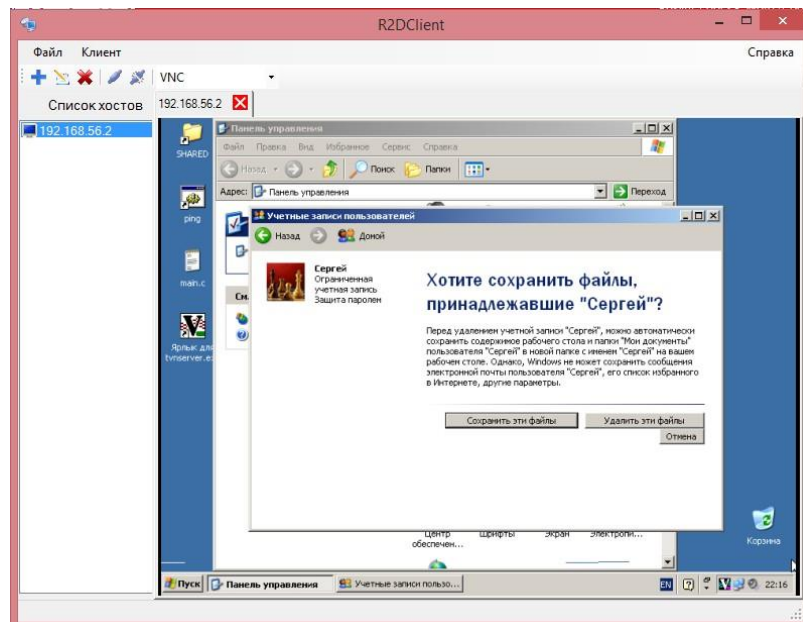


Рис. 2.14. Видалення запису користувача на клієнтському додатку

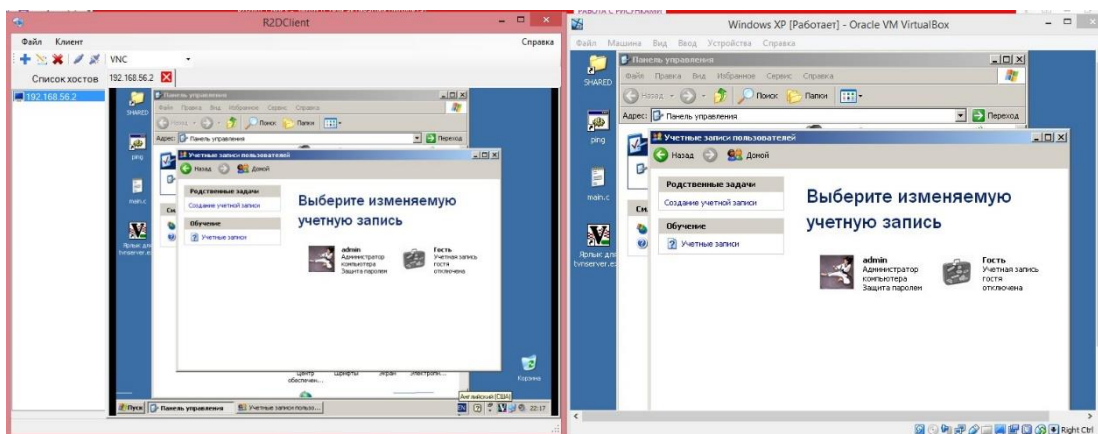


Рис. 2.15. Результаты віддаленого адміністрування

4. Підключення до хосту, дані про який зберігаються в файлі (\* .rdp). В першу чергу створимо rdp-файл через вікно Підключення до віддаленого робочого столу. Зміст rdp-файлу наведено у лістингу нижче. IP - адреса віддаленого комп'ютера виділена жирним шрифтом.

Лістинг 2.9

```
screen mode id:i:2
use multimon:i:0
desktopwidth:i:1600
desktopheight:i:900
session bpp:i:32
winposstr:s:0,3,0,0,800,600
compression:i:1
keyboardhook:i:2
audiocapturemode:i:0
videoplaybackmode:i:1
connection type:i:7
networkautodetect:i:1
bandwidthautodetect:i:1
displayconnectionbar:i:1
enableworkspacereconnect:i: 0
disable wallpaper:i:0
allow font smoothing:i:0
allow desktop composition:i:0
disable full window drag:i:1
disable menu anims:i:1
disable themes:i:0
disable cursor setting:i:0
bitmapcachepersistenable:i:1
full
address:s:192.168.56.2
```

audiomode:i:0  
redirectprinters:i:1  
redirectcomports:i:0  
redirectsmartcards:i:1  
redirectclipboard:i:1  
redirectposdevices:i:0  
drivestoredirect:s:  
autoreconnection  
enabled:i:1 authentication  
level:i:2 prompt for  
credentials:i:0 negotiate  
security layer:i:1  
remoteapplicationmode:i:0

Виконаємо команду Файл-Відкрити на клієнтському додатку. Відкриється вікно діалогу для вибору rdp-файлу (рис. 2.16). Після вибору файлу (Default.rdp) в панелі Список хостів клієнтського додатку з'явиться IP-адреса віддаленого комп'ютера, після чого можна виконувати операцію з'єднання (рис. 2.17).

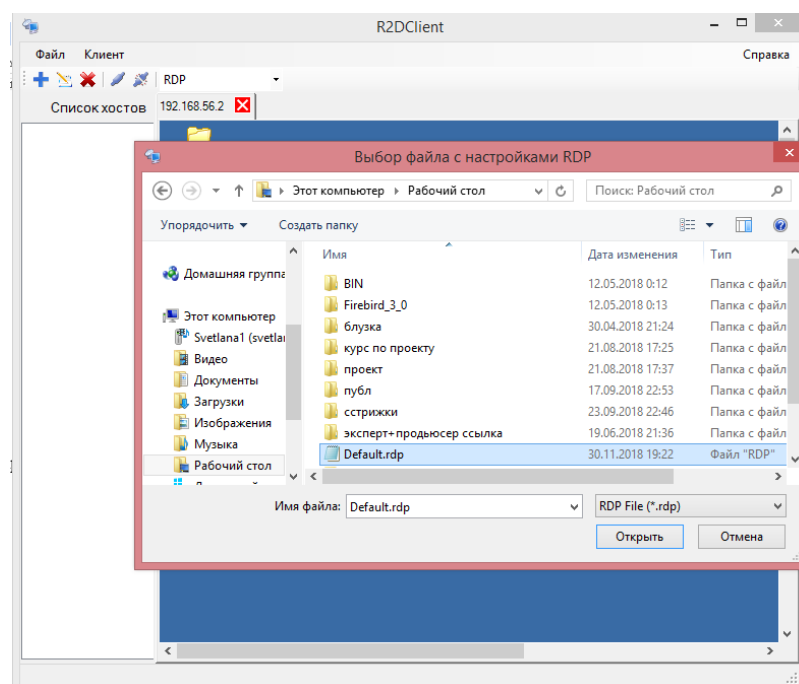


Рис. 2.16. Вікно діалогу для вибору rdp-файлу

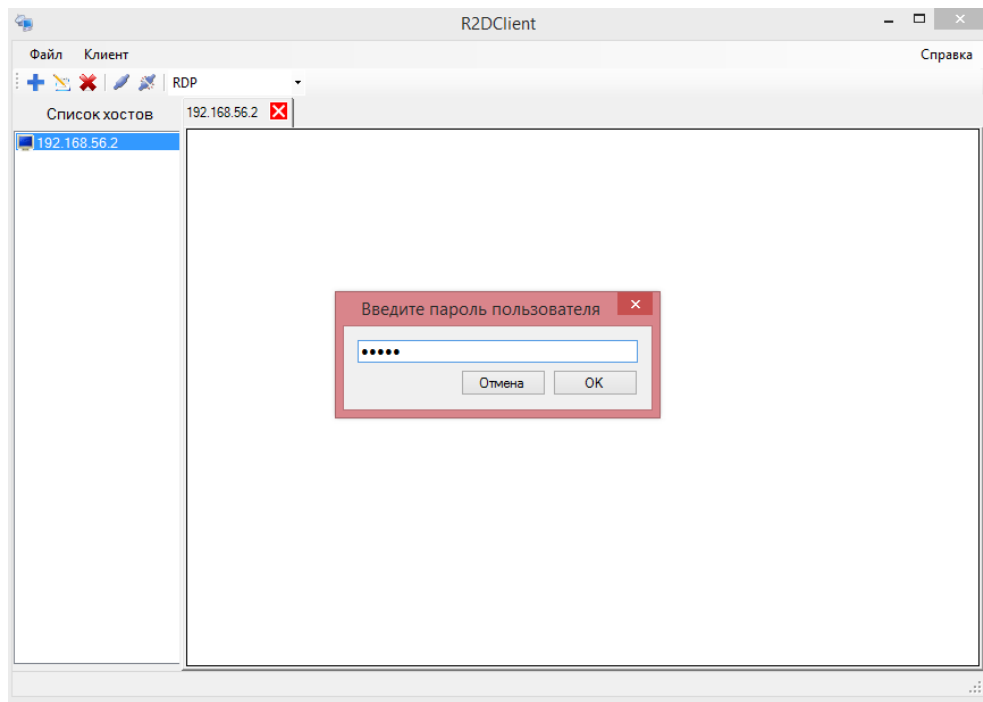


Рис. 2.17. Вікно з'єднання з віддаленим комп'ютером за допомогою rdp-файлу

Висновок: за результатами тестування розроблений додаток повністю відповідає усім цілям та вимогам.

### Висновок до розділу

У другому розділі магістерської роботи було обґрунтовано інструменти розробки додатку, а саме: обрано мову програмування C# і середовище розробки Microsoft Visual Studio 2010; для реалізації інтерфейсної частини додатку застосовується бібліотека Windows Forms на основі технології .Net Framework 4; в реалізації моделі доступу до даних застосовується технологія ADO.NET.

Також сформульовано основні цілі та вимоги до додатку:

- здійснювати віддалений доступ за протоколами RDP і VNC.
- підключатися до хосту, дані про який зберігаються в файлі (\*.rdp).
- спостерігати і керувати віддаленим комп'ютером в мережі.
- зберігати дані про вузол підключення в базі даних.

Для реалізації клієнта було вирішено взяти за основу трирівневу архітектуру, кожен шар якої буде відповідати за свою логічну частину: рівень

відображення, рівень логіки - управління підключенням, рівень протоколу.

Для реалізації можливості зберігання даних про хости, що підключаються, була використана база даних SQLite на базі технології доступу до даних ADO.NET.

Таким чином, було розроблено клієнтський додаток віддаленого доступу на платформі .Net, який відповідає всім встановленим цілям і вимогам.

## ВИСНОВКИ

Під час виконання магістерської роботи було проведено аналіз технологій віддаленого доступу та розроблено програмний додаток віддаленого керування робочою станцією на платформі .NET

Встановлено, що віддалений доступ являє собою функцію, що дозволяє користувачеві підключатися до комп'ютера через Інтернет за допомогою іншого ПК. Одним з видів віддаленого доступу є доступ до віддаленого робочого столу (Remote Desktop), доступ до якого здійснюється на базі архітектури «сервер - термінал».

Проаналізовано принципи роботи протоколів віддаленого доступу: RDP, VNC, ICA. На підставі проведеного аналізу було вирішено обрати для розробки проекту протоколи RDP і VNC, оскільки їх застосування дозволить реалізувати віддалений доступ не тільки до комп'ютерів з операційною системою Windows, а також дасть можливість застосовувати різні платформи на клієнті і сервері віддаленого доступу.

Проведений аналіз додатків (TeamViewer, LiteManager, Ammy admin, RAdmin) який довів, що програми, що забезпечують віддалений доступ та віддалене адміністрування, потребують наявності (придбання) ліцензії для довготривалої роботи з програмою. Тому розробка власного додатку віддаленого доступу може вирішити цю проблему.

Під час розробки клієнтського додатку було обґрунтовано інструменти розробки, а саме: обрано мову програмування C # і середовище розробки Microsoft Visual Studio 2010; для реалізації інтерфейсної частини додатку застосовується бібліотека Windows Forms на основі технології .Net Framework 4; в реалізації моделі доступу до даних застосовується технологія ADO.NET.

Також сформульовано основні цілі та вимоги до додатку: здійснювати віддалений доступ за протоколами RDP і VNC; підключатися до хосту, дані про який зберігаються в файлі (\* .rdp); спостерігати і керувати віддаленим комп'ютером в мережі; зберігати дані про вузол підключення в базі даних.

Для реалізації клієнта було вирішено взяти за основу трирівневу

архітектуру: рівень відображення, рівень логіки - управління підключенням, рівень протоколу.

Для реалізації можливості зберігання даних про хости була використана база даних SQLite на базі технології доступу до даних ADO.NET.

Таким чином, було розроблено програмний додаток віддаленого керування робочою станцією на платформі .Net, який відповідає всім встановленим цілям і вимогам.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. TL-WR840N V6.20 URL: <https://www.tp-link.com/uk-ua/home-networking/wifi-router/tl-wr840n/> (дата звернення: 21.12.2022) 12.AC1200 Двухдиапазонный гигабитный Wi-Fi роутер URL: <https://www.mercusys.com/ru/product/details/ac12g> (дата звернення: 21.12.2023)
2. AX1500 Wi-Fi 6 маршрутизатор URL: <https://www.tp-link.com/uk-ua/home-networking/wifi-router/archer-ax10/> (дата звернення: 21.12.2023)
3. Маршрутизатор MikroTik hEX (RB750Gr3) URL: <https://www.mikrotik.ua/product/mikrotik-hex-rb750gr3> (дата звернення: 21.12.2023)
4. Каталог приладів MikroTik URL <https://xn----7sba7aachdbqfnhtigr1.xn--j1amh/katalog-ustrojstv/nastrojka-mikrotik-hex-rb750gr3/> (дата звернення: 21.12.2023)
5. RouterOS URL: <https://xn----7sba7aachdbqfnhtigr1.xn--j1amh/kategoriya-ustroystva/vse-kategorii/routeros/> (дата звернення: 21.12.2023)
6. Налаштування VPN через MikroTik URL: <https://deps.ua/ua/knowegable-base/samples-of-the-technical-solutions/7990.html> (дата звернення: 21.12.2023)
7. Вилдермьюс Ш. Практическое использование ADO.NET. Доступ к данным в Internet. / Вилдермьюс, Шон. : Пер. с англ. — М. : Издательский дом "Вильяме", 2003. — 288 с.
8. Виртуализация Citrix XenServer, XenDesktop и XenApp [Электронный ресурс]. – Режим доступа к ресурсу: <http://www.vmgw.ru/citrix-xen>.
9. Ибе О. Сети и удаленный доступ. Протоколы, проблемы, решения / Оливер Ибе. – Пер. И. Сеницын. — М.: ДМК Пресс, 2002. — 336 с.
10. Калюжный А.Д. Анализ средств сжатия видеосигнала / А.Д. Калюжный, Г.В. Табунщик // Системи обробки інформації: зб. наук. пр. – Х.: ХУПС, 2010. – Вип. 7(88). – С. 200.
11. Мазерс Т.В. Архитектура тонкого клиента в WindowsNT/2000.

- Реализация терминальных служб и Citrix MetaFrame / Т.В. Мазерс. – М.: Вильямс, 2001. – 800 с.
12. Норма: Терминальные решения [Электронный ресурс]. – Режим доступа к ресурсу: [http://www.norma-ts.ru/support\\_info\\_04.shtml](http://www.norma-ts.ru/support_info_04.shtml).
13. Переход на терминальные системы в условия экономического кризиса [Электронный ресурс]. – Режим доступа к ресурсу: [http://omsk.narod.ru/documents/5\\_prichin.html](http://omsk.narod.ru/documents/5_prichin.html)
14. Принцип работы Microsoft Terminal Services [Электронный ресурс]. Режим доступа: [http://technet.microsoft.com/en-us/library/cc755399\(W.S.10\).aspx](http://technet.microsoft.com/en-us/library/cc755399(W.S.10).aspx)
15. Решения для удаленной работы с домашним компьютером. [Электронный ресурс]. – Режим доступа: <http://compress.ru/> (дата обращения: 28.11.2016).
16. Рихтер Дж. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C# / Рихтер Дж. —4-е изд. — СПб.: Питер, 2013. — 896 с
17. Робинсон С. C# для профессионалов / Робинсон С., Корнес О., Глинн Д. и др. — Изд-во: "Лори", Москва, 2003. — 1024 с.
18. Сеть и удаленный доступ к сети [Электронный ресурс]. – Режим доступа к ресурсу: <http://mif.vspu.ru/books/w2k/gl16/gl16.html>
19. Системы удаленного управления. [Электронный ресурс]. – Режим доступа: <http://www.osp.ru/> (дата обращения: 28.11.2016).
20. Создание и настройка виртуальной машины в VirtualBox [Электронный ресурс]. – Режим доступа к ресурсу: <https://www.gotoadm.ru/create-and-settings-virtual-machine-in-virtualbox/>
21. Терминальные решения [Электронный ресурс]. – Режим доступа к ресурсу: [http://www.onix.kiev.ua/terminal\\_advantages.asp](http://www.onix.kiev.ua/terminal_advantages.asp)
22. Тонкие клиенты и терминальные системы Citrix и Windows [Электронный ресурс]. – Режим доступа к ресурсу: [http://www.uw.ru/thin\\_client/](http://www.uw.ru/thin_client/).

23. Удаленный доступ к ПК без риска // СНР. — 2015. — №12. — [Электронный ресурс]. — Режим доступа к ресурсу: <https://ichip.ru/category/podborki>
24. Управление «Институт информатики ИжГТУ» [Электронный ресурс]. — Режим доступа к ресурсу: <http://www.pro-spo.ru/index.php>
25. Шилдт Г. С#: Учебный курс / Шилдт Г. — СПб.: Питер, 2003. — 512 с.
26. Черкасова Н.И., Сетевые операционные системы. Сетевые файловые системы и служба каталогов. Учебное пособие. [Текст] / Н.И. Черкасова — М.: МГТУ ГА, 2010. — 68 с.
27. RAdmin — руководство по продажам [Электронный ресурс]. — Режим доступа: <http://old.mont.ru/docs/> (дата обращения: 28.11.2016).
28. Remote Desktop Protocol Performance [Электронный ресурс]. — Режим доступа к ресурсу: [http://download.microsoft.com/download/4/d/9/4d9ae285-3431-4335-a86e-969e7a146d1b/rdp\\_performance\\_whitepaper.docx](http://download.microsoft.com/download/4/d/9/4d9ae285-3431-4335-a86e-969e7a146d1b/rdp_performance_whitepaper.docx).
29. Security Lab by Positive Technologies [Electronic resource]. — Resource Access Mode <http://www.securitylab.ru/analytics/367591.php>
30. The RFB Protocol [Electronic resource]. — Resource Access Mode: <http://www.tigervnc.com/cgi-bin/rfbproto#tight-security-type>.
31. The Trusted Solution for Remote Desktop Control [Electronic resource]. — Resource Access Mode: <https://www.teamviewer.com/en/solutions/remote-desktop/#gref>
32. VNC Virtual Network Computing [Electronic resource]. — Режим доступа к ресурсу: <https://www.raspberrypi.org/documentation/remote-access/vnc/>
33. Windows Shop [Electronic resource]. Resource Access Mode: <http://www.microsoft.com/windows/buy/default.aspx>
34. What is a VNC (Virtual Network Computing)? [Electronic resource]. — Resource Access Mode: <http://www.remoteaccess.org/what-is-a-vnc/>

## ДОДАТОК

### Лістинг файлу MainWindow.cs

```
using System;
using System.Collections.Generic; using System.ComponentModel; using
System.Data;
using System.Drawing; using System.Linq; using System.Text;
using System.Windows.Forms; using RDPFileReader;
using VncSharp;
namespace R2DClient
{
    public partial class MainWindow : Form
    {
        private RDPSettingsDialog RDPSettingsDlg = new RDPSettingsDialog();
        private SQLiteInterface SQL = new SQLiteInterface(); private
List<RemoteAbstractWidget> widgets;
        public MainWindow()
        {
            InitializeComponent();
            widgets = new List<RemoteAbstractWidget>();
            cbVariantClient.SelectedIndex = 0;
            closeMenu();
            RDPSettingsDlg.setSize(Convert.ToInt32(SettingsIni.IniReadValue("RDP",
"size"))));
            RDPSettingsDlg.setColor(Convert.ToInt32(SettingsIni.IniReadValue("RDP"
, "color"))));
            SQL.connectionString = "Data Source=hosts.db; Version=3;";
            SQL.addTable("hostdata");
            SQL.fill("hostdata");
            DataTable table = SQL.dataTable("hostdata"); for (int i = 0; i <
table.Rows.Count; ++i)
```

```

ipList.Items.Add(new IListBoxItem(table.Rows[i][1].ToString(), 0));
ipList.SelectedIndex = 0;
}
private void settingsRDPMenuItem_Click(object sender, EventArgs e)
{
    RDPSettingsDlg.setSize(Convert.ToInt32(SettingsIni.IniReadValue("RDP",
"size")));
    RDPSettingsDlg.setColor(Convert.ToInt32(SettingsIni.IniReadValue("RDP",
, "color")));
    RDPSettingsDlg.ShowDialog();
}
private void deleteMenuItem_Click(object sender, EventArgs e)
{
    DataTable table = SQL.dataTable("hostdata");
table.Rows[ipList.SelectedIndex].Delete(); SQL.save("hostdata");
ipList.Items.RemoveAt(ipList.SelectedIndex);
}
private void deleteHostBotton_Click(object sender, EventArgs e)
{
    deleteMenuItem_Click(sender, e);
}
private void addHostMenuItem_Click(object sender, EventArgs e)
{
    AddHostDialog HostDlg = new AddHostDialog(SQL);
HostDlg.ShowDialog();
    string newHost = HostDlg.getNewHost(); if (newHost != string.Empty)
ipList.Items.Add(new IListBoxItem(newHost, 0));
}
private void addHostButton_Click(object sender, EventArgs e)
{

```

```

addHostMenuItem_Click(sender, e);
}

private void editHostStripMenuItem_Click(object sender, EventArgs)
{
    AddHostDialog HostDlg = new AddHostDialog(SQL); DataTable table =
SQL.dataTable("hostdata"); DataRow row = table.Rows[ipList.SelectedIndex];
HostDlg.ShowDialog(row);

    string newHost = HostDlg.getNewHost(); if (newHost != string.Empty)
    ipList.Items[ipList.SelectedIndex]=new
    IListItem(newHost, 0);
}

private void openHostEdit(object sender, EventArgs e)
{
    editHostStripMenuItem_Click(sender, e);
}

private void editHostBotton_Click(object sender, EventArgs e)
{
    editHostStripMenuItem_Click(sender, e);
}

private void rdpConnectMenuItem_Click(object sender, EventArgs e)
{
    RemoteRDPCClient client = new RemoteRDPCClient(); widgets.Add(client);
   TabPage NewPage = new TabPage();
    NewPage.Text= Convert.ToString(ipList.Items[ipList.SelectedIndex]);
    NewPage.Controls.Add(client.Widget);
tabControl.TabPages.Add(NewPage); tabControl.SelectedIndex =
tabControl.TabCount - 1;

    DataTable table = SQL.dataTable("hostdata"); DataRow row =
table.Rows[ipList.SelectedIndex];
    if (client.setData(row))

```

```

    {
        closeMenu(client); client.setColorDepth(RDPSettingsDlg.getColorDepth());
client.setDesktopSize(RDPSettingsDlg.getDesktopSize()); client.connect();
    }
    else disconnectButton_Click(sender, e);
    }

    private void vncConnectMenuItem_Click(object sender, EventArgs e)
    {
        RemoteVNCCClient client = new RemoteVNCCClient(); widgets.Add(client);
       TabPage NewPage = new TabPage();
        NewPage.Text= Convert.ToString(ipList.Items[ipList.SelectedIndex]);
NewPage.Controls.Add(client.Widget); tabControl.TabPages.Add(NewPage);
        tabControl.SelectedIndex = tabControl.TabCount - 1;
        DataTable table = SQL.dataTable("hostdata"); DataRow row =
table.Rows[ipList.SelectedIndex];
        if (client.setData(row))
        {
            closeMenu(client); client.connect();
        }
        else
            disconnectButton_Click(sender, e);
    }

    private void openMenuItem_Click(object sender, EventArgs e)
    {
        OpenFileDialog myOpenFileDialog; myOpenFileDialog = new
OpenFileDialog(); myOpenFileDialog.Filter = "RDP File|*.rdp";
        myOpenFileDialog.Title = "Выбор файла с настройками RDP";
myOpenFileDialog.ShowDialog();
        if (myOpenFileDialog.FileNames.Count() > 0)
        {

```

```

RDPFile MyRdpFile = new RDPFile();
MyRdpFile.Read(myOpenFileDialog.FileName);
    if (MyRdpFile.FullAddress != string.Empty)
    {
        DataRow row = SQL.newRow("hostdata");
        row["ip"] = MyRdpFile.FullAddress; row["port"] = string.Empty;
row["login"] = MyRdpFile.Username;
        row["password"] = Helper.XorString(MyRdpFile.Password,
SQL.addNewRow("hostdata", row);
        ipList.Items.Add(new IListItem(MyRdpFile.FullAddress,
    }
    }

private void closeTab(object sender, ClosingEventArgs e)
{
    if (tabControl.SelectedIndex >= 0)
    {
        RemoteAbstractWidgetaWidget= widgets[tabControl.SelectedIndex];
        aWidget.disconnect(); widgets.RemoveAt(tabControl.SelectedIndex);
tabControl.TabPages.RemoveAt(tabControl.SelectedIndex);
    }
}

private void disconnectButton_Click(object sender, EventArgs e)
{
    if (tabControl.SelectedIndex >= 0)
    {
        RemoteAbstractWidgetaWidget= widgets[tabControl.SelectedIndex];
        aWidget.disconnect(); widgets.RemoveAt(tabControl.SelectedIndex);
tabControl.TabPages.RemoveAt(tabControl.SelectedIndex);
    }
    if (tabControl.SelectedIndex != -1)

```

```

{
    RemoteAbstractWidget aWidget = widgets[tabControl.SelectedIndex];
    closeMenu(aWidget);
}
else closeMenu();
}

private void connectBotton_Click(object sender, EventArgs e)
{
    if (cbVariantClient.SelectedIndex == 0) rdpConnectMenuItem_Click(sender,
e);

    if (cbVariantClient.SelectedIndex == 1)
vncConnectMenuItem_Click(sender, e);
}

private void ctrl_alt_delVNCMenuItem_Click(object sender,
EventArgs e)
{
    RemoteAbstractWidget aWidget = widgets[tabControl.SelectedIndex];
    if (aWidget.isClient() == "VNC")
    {
        RemoteVNCClient client = (RemoteVNCClient)aWidget;
        if (client.Widget.IsConnected)
client.Widget.SendSpecialKeys(SpecialKeys.CtrlAltDel);
    }
}

private void ctrl_escVNCMenuItem_Click(object sender, EventArgs e)
{
    RemoteAbstractWidget aWidget =
widgets[tabControl.SelectedIndex];
    if (aWidget.isClient() == "VNC")
    {

```

```

        RemoteVNCCClient client = (RemoteVNCCClient)aWidget; if
(client.Widget.IsConnected)
        client.Widget.SendSpecialKeys(SpecialKeys.CtrlEsc);
    }
}

private void alt_f4VNCMMenuItem_Click(object sender, EventArgs e)
{
    RemoteAbstractWidget aWidget= widgets[tabControl.SelectedIndex];
    if (aWidget.isClient() == "VNC")
    {
        RemoteVNCCClient client = (RemoteVNCCClient)aWidget; if
(client.Widget.IsConnected)
        client.Widget.SendSpecialKeys(SpecialKeys.AltF4);
    }
}

private void ctrlVNCMMenuItem_Click(object sender, EventArgs e)
{
    RemoteAbstractWidget aWidget= widgets[tabControl.SelectedIndex];
    if (aWidget.isClient() == "VNC")
    {
        RemoteVNCCClient client = (RemoteVNCCClient)aWidget; if
(client.Widget.IsConnected)
        client.Widget.SendSpecialKeys(SpecialKeys.Ctrl, false/* don't release CTRL
*/);
    }
}

private void altVNCMMenuItem_Click(object sender, EventArgs e)
{
    RemoteAbstractWidget aWidget=
widgets[tabControl.SelectedIndex];

```

```

        if (aWidget.isClient() == "VNC")
        {
            RemoteVNCClient client = (RemoteVNCClient)aWidget; if
(client.Widget.IsConnected)
                client.Widget.SendSpecialKeys(SpecialKeys.Alt,false/* don't release ALT
*/);
        }
    }

    private void copyClipboardMenuItem_Click(object sender, EventArgs e)
    {
        RemoteAbstractWidget aWidget= widgets[tabControl.SelectedIndex];
        if (aWidget.isClient() == "VNC")
        {
            RemoteVNCClient client = (RemoteVNCClient)aWidget; if
(client.Widget.IsConnected)
            {
                client.Widget.FillServerClipboard();
                локальный хост",
                MessageBox.Show(this, "Буффер обмена скопированн на
                "Копирование буффера обмена", MessageBoxButtons.OK,
                MessageBoxIcon.Information);
            }
        }
    }

    private void closeMenu(RemoteAbstractWidget aWidget)
    {
        if (aWidget.isClient() == "RDP")
        {
            vncMenuItem.Enabled = false; sendKeysVNCMenuItem.Enabled = false;
            ctrl_alt_delVNCMenuItem.Enabled = false;

```

```

        ctrl_escVNCMenuItem.Enabled = false; alt_f4VNCMenuItem.Enabled =
false; ctrlVNCMenuItem.Enabled = false; altVNCMenuItem.Enabled = false;
copyClipboardMenuItem.Enabled = false;

        rdpMenuItem.Enabled = true; settingsRDPMMenuItem.Enabled = true;
        cbVariantClient.SelectedIndex = 0;
    }

    if (aWidget.isClient() == "VNC")
    {
        vncMenuItem.Enabled = true; vncMenuItem.Enabled = true;
sendKeysVNCMenuItem.Enabled = true; ctrl_alt_delVNCMenuItem.Enabled =
true; ctrl_escVNCMenuItem.Enabled = true; alt_f4VNCMenuItem.Enabled = true;
ctrlVNCMenuItem.Enabled = true; altVNCMenuItem.Enabled = true;
copyClipboardMenuItem.Enabled = true;

        rdpMenuItem.Enabled = false; settingsRDPMMenuItem.Enabled = false;
        cbVariantClient.SelectedIndex = 1;
    }
}

private void closeMenu()
{
    vncMenuItem.Enabled = false; sendKeysVNCMenuItem.Enabled = false;
ctrl_alt_delVNCMenuItem.Enabled = false; ctrl_escVNCMenuItem.Enabled =
false; alt_f4VNCMenuItem.Enabled = false; ctrlVNCMenuItem.Enabled = false;
altVNCMenuItem.Enabled = false; copyClipboardMenuItem.Enabled = false;
rdpMenuItem.Enabled = true; settingsRDPMMenuItem.Enabled = true;
}

private void selectTab(object sender, TabControlEventsArgs e)
{
    if (tabControl.SelectedIndex != -1)
    {
        RemoteAbstractWidget aWidget= widgets[tabControl.SelectedIndex];

```

```

        closeMenu(aWidget);
        if (aWidget.isClient() == "RDP"){
            RemoteRDPCClient client = (RemoteRDPCClient)aWidget;
tabControl.TabPages[tabControl.SelectedIndex].Update(); client.Widget.Update();
        }
        if (aWidget.isClient() == "VNC"){
            RemoteVNCCClient client = (RemoteVNCCClient)aWidget;
client.Widget.Update();
        }
        }
        else closeMenu();
    }

    private void aboutMenuItem_Click(object sender, EventArgs e)
    {
        AboutBox box = new AboutBox(); box.ShowDialog();
    }

    private void exitMenuItem_Click(object sender, EventArgs e)
    {
        for (int i = 0; i < tabControl.TabCount; ++i) disconnectButton_Click(sender,
e);
        Close();
    }

    private void tabControl_SelectedIndexChanged(object sender,
EventArgs e)
    {
    }
    }
    }
    }

```