МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ

ТЕРНОПІЛЬСЬКИЙ_____

КАФЕДРА _____

Допускається до захисту

завідувач кафедри КБ

_____ _____

ПОЯСНЮВАЛЬНА

ЗАПИСКА

до дипломного проекту на тему:

«РОЗРОБЛЕННЯ МОДУЛЯ ЗАБЕЗПЕЧЕННЯ ПРИХОВУВАННЯ ДАНИХ НА ОСНОВІ СТЕГАНОГРАФІЧНОГО МЕТОДУ НАЙМЕНШ ЗНАЧУЩОГО БІТУ»

освітньо-кваліфікаційний рівень – магістер

Студент ____ гр. _____

Керівник проекту

_____ _____

_____

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСТКИЙ НАЦІОНАЛЬНИЙ _____

Факультет _____
Кафедра _____
Спеціальність «_____»

ЗАТВЕРДЖУЮ:
завідувач кафедри КБ
_____
_____.
«__» _____ 20____ р.

ЗАВДАННЯ
на магістерську роботу студента
_____

1. Тема роботи: «Розроблення модуля забезпечення приховування даних на основі стеганографічного методу найменш значущого біту» затверджена наказом по університету від «___» _____ 20__ р. № ____

2. Термін здачі студентом закінченої роботи: _____

3. Вхідні дані до роботи: Передбачити в роботі аналіз стану досліджень по вдосконаленню мережі Інтернет, розглянути архітектуру семантичної «павутини», напрямки використання програмних агентів в мережі Інтернет, існуючі підходи до побудови програмних агентів в семантичній «павутині», дослідити технології використання програмних агентів в семантичній «павутині», розробити методику дослідження ефективності використання пошукових програмних агентів, виконати аналіз показників ефективності використання пошукового програмного агенту. В процесі розробки програмного засобу використати мову програмування C# та середовище MS Visual Studio 2008, технологію ASP.NET.

4. Зміст пояснювальної записки (перелік питань, що підлягають розробленню): Вступ. 1. Аналіз сучасного стану досліджень щодо вдосконалення мережі Інтернет. 2 Використання пошукових програмних агентів в семантичній «павутині». 3. Дослідження ефективності використання пошукового програмного агенту в семантичній «павутині». Висновки. Додатки.

5. Консультанти по проекту із зазначенням розділів проекту, що їх стосуються

Найменування розділу	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання роботи	Термін виконання етапів	Примітки
1	Аналіз сучасного стану досліджень щодо вдосконалення мережі Інтернет		
2	Використання пошукових програмних агентів в семантичній «павутині»		
3	Дослідження ефективності використання пошукового програмного агенту в семантичній «павутині»		
4	Підготовка пояснювальної записки		
5	Підготовка презентації та доповіді		
6	Попередній захист		
7	Нормоконтроль, рецензування		
8	Занесення диплома в електронний архів		
9	Допуск до захисту у зав. кафедрою		

Дата видачі завдання «__» _____ 20___ р.

Керівник к.т.н., доц. кафедри КБ_____ _____
(підпис)

Завдання прийняв до виконання _____ _____
(підпис)

АНОТАЦІЯ

Розроблення модуля забезпечення приховування даних на основі стеганографічного методу найменш значущого біту // Дипломна робота ОР «Магістр» // Кучма Олександр Русланович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра кібербезпеки, група СБм-61 // Тернопіль, 2019 // С. 93, рис. – 34 , табл. – 26 , додат. – 5.

Ключові слова: СТЕГАНОГРАФІЯ, МЕТОД НАЙМЕНШ ЗНАЧУЩОГО БІТУ, ЦИФРОВИЙ ВОДЯНИЙ ЗНАК, ПРИХОВУВАННЯ ІНФОРМАЦІЇ В ЦИФРОВИХ ЗОБРАЖЕННЯХ, CASE-ДІАГРАМИ, БАЗА ДАНИХ, СТЕГАНОКОНТЕЙНЕР, СТЕГАНОСИСТЕМА.

Пояснювальна записка містить результати розроблення модуля «Вбудовування інформації в зображенні методом найменш значущого біту». Проведено аналіз предметної області, розроблені моделі бізнес-процесів об'єкта проектування, розроблені специфікації бізнес-вимог, функціональних та не функціональних вимог, спроектовані логічна та фізична моделі бази даних. Виконаний математичний опис задачі для вирішення на ПК та розроблений додаток за допомогою середовища програмування Microsoft Visual Studio 2010. Наведені результати тестування програмного продукту та вимоги щодо його розгортання. Результати роботи можуть бути використані для підтвердження авторського права, для приховування інформації в цифрових зображеннях.

ANOTATION

Development of a data hiding module based on steganographic method of the least significant bit // Thesis of the Master degree // Oleksandr Kuchma // Ternopil Ivan Puluj National Technical University, Department of Computer Information Systems and Software Engineering, Department of Cybersecurity // Ternopil, 2019 // P. 93, Fig. – 34, Tables – 26, Annexes. – 5.

Keywords: STEGANOGRAPHY, ENCRYPTION, INFORMATION PROTECTION TECHNIQUES, THE LEAST SIGNIFICANT BIT, DIGITAL WATERMARKING, WATERMARK, HIDING INFORMATION IN DIGITAL IMAGES, CASE-CHART, DATABASE

Explanatory note contains the results of the development module "Embedding information in an image using the least significant bit. The analysis domain, the model business processes of the project, developed the specification of business requirements, functional and non functional requirements, designed logical and physical database models. The mathematical description of the problem for solution on the PC application developed using the programming environment Microsoft Visual Studio 2010. The results of software testing and requirements for its deployment. The results may be used to confirm the copyright for hiding information in digital images.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

BPSK – застосовуватися двійкова відносна фазова модуляція
ПСП – псевдо випадкова послідовність
ЦВЗ – цифрові водяні знаки
LSB – метод найменш значущих бітів
ПЗ – програмного забезпечення
ЦС – цифрова стеганографія

ВСТУП

Комп’ютерні технології надали нового імпульсу розвитку і вдосконаленню стеганографії, з’явився новий напрямок в галузі захисту інформації – цифрова стеганографія (ЦС) [3].

Ці методи, враховуючи природні неточності пристроїв оцифровування і надмірність аналогового відео або аудіо сигналу, дозволяють приховувати повідомлення в комп’ютерних файлах (контейнерах). Причому, на відміну від криптографії, дані методи приховують сам факт передачі інформації.[5]

Клод Шеннон дав загальну теорію тайнопису, яка є базисом стеганографії як науки [11].

Основні положення цифрової стеганографії є:

методи приховування повинні забезпечувати автентичність і цілісність файлу;

передбачається, що аналітику повністю відомі можливі стеганографічні методи;

безпека методів ґрунтується на збереженні стеганографічних перетворенням основних властивостей відкритого файлу, що передається при внесенні до нього таємного повідомлення і деякої невідомої супротивникові інформації – ключа;

навіть якщо факт приховування повідомлення став відомий зловмиснику через спільника, витяг самого секретного повідомлення являє складну обчислювальну задачу.

Основним завданням стеганографії є подолання систем моніторингу та управління мережевими ресурсами. Стеганографічні методи, спрямовані на протидію систем моніторингу та управління мережевими ресурсами промислового шпигунства, дозволяють протистояти спробам контролю над інформаційним простором при проходженні інформації через сервери управління локальних і глобальних обчислювальних мереж. Іншим важливим завданням стеганографії є камуфлюванні програмного забезпечення (ПЗ). У тих випадках, коли використання ПЗ незареєстрованими користувачами є небажаним, воно може бути закамуфльоване під стандартні універсальні програмні продукти (наприклад, текстові редактори) або приховано у файлах мультимедіа (наприклад, у звуковому супроводі комп’ютерних ігор). Ще однією областю використання стеганографії є захист авторського права від піратства. На комп’ютерні графічні зображення наноситься спеціальна позначка, яка залишається невидимою для очей, але розпізнається спеціальним

ПЗ. Таке програмне забезпечення вже використовується в комп'ютерних версіях деяких журналів. Даний напрямок стеганографії призначене не тільки для обробки зображень, але й для файлів з аудіо і відео наповненням покликане забезпечити захист інтелектуальної власності.

В даній роботі детальніше розглянемо метод найменш значущих бітів (Least Significant Bit, LSB). Цей метод є найбільш поширеним в цифровій стеганографії. З'явився на початку 90-х років 20-го століття, він ґрунтується на обмежених здібностях органів почуттів, внаслідок чого людям дуже важко розрізняти незначні варіації звуку та кольору. Розглянемо цей метод на прикладі 24 бітного растрового RGB зображення. Кожна точка кодується трьома байтами, кожен байт визначає інтенсивність червоного (Red), зеленого (Green) і синього (Blue) кольору. Сукупність інтенсивностей кольору в кожному з трьох каналів визначає відтінок піксела. Науково підтверджено факт, що система людського зору найменш чутлива до змін інтенсивності у синій області спектра. Таким чином можна з великою впевненістю підміняти молодший біт байта, що відповідає за інтенсивність синього каналу, по обраній нами закономірності. Людському оку буде принципово важко відрізнити оригінальне чисте зображення від стеганопосилки. А у відсутності оригіналу завдання виявлення змін стає практично неможливою.

Таким чином на сьогоднішній день існують механізми захисту конфіденційних даних які не забезпечують прихованість факту передачі даних в телекомунікаційних і інформаційних системах. Тому актуальною стає задача розроблення модуля забезпечення приховування даних на основі стеганографічного методу найменш значущого біту.

РОЗДІЛ 1

1.1 Коротка характеристика підприємства

Базою підприємства виступає Інтернет-магазин «TradeBox».

Базою підприємства був обраний суб'єкт підприємницької діяльності. Основним видом його діяльності є надання послуг по купівлі клієнтами побутової та офісної техніки через Інтернет. Фірма має розвинену структуру й складається з відділів. Очолює фірму директор. Йому підпорядковується секретар-референт та наступні відділи:

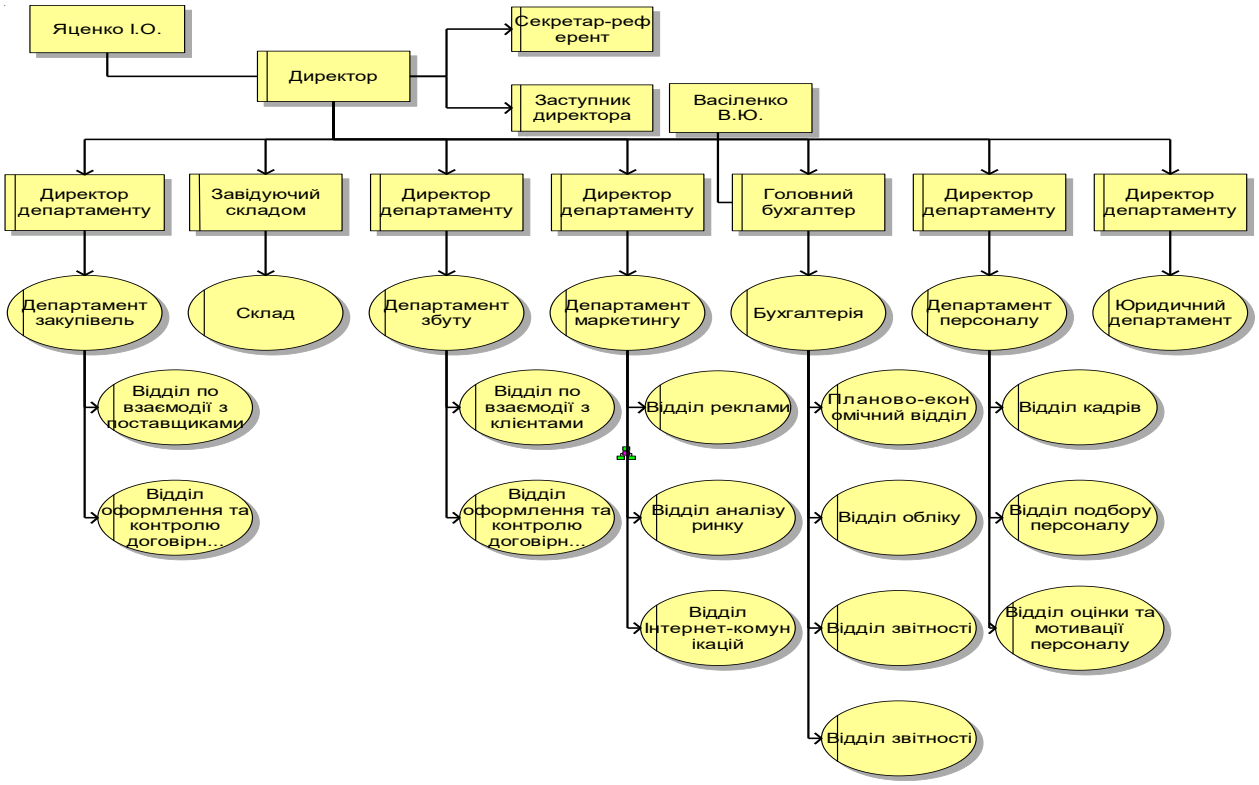


Рис 1.1. Загальна організаційна схема підприємства «TradeBox»

департамент закупівель;

департамент маркетингу;

юридичний департамент;

департамент збуту;

департамент персоналу;

IT-департамент;

склад;

бухгалтерія.

Загальна організаційна схема підприємства виконана в пакеті ARIS 6.2 і показана на рис. 1.1.

Основні напрямки діяльності підприємтсва є:

використання й впровадження інформаційних технологій;

створення й розповсюдження рекламних даних;

аналізування та вивчення попиту населення, щодо купівлі побутової та офісної техніки за період з 2006 по 2009 рр;

зіставлення плану розвитку підприємтва на основі даних аналізу та урахуванням споживчої здатності населення на наступний період з 2010 по 2013 рр;

Основними видами діяльності є:

закупівля товару від оптових постачальників;

доставка товару за місцем призначення;

установлення товару за бажанням замовника;

розміщення нових видів товару на сайті підприємства;

консультація та кредитування населення через банки, з якими співпрацює підприємство;

заключення договорів на постачання товару;

поліграфічна продукція.

1.2 Аналіз предметної області

Цифрова стеганографія як наука появилася недавно. Вона має наступні напрями[8]:

вбудовування інформації з метою її прихованого передавання;

вбудовування оцифрованих водяних знаків (watermarking);

вбудовування номерів ідентифікації (fingerprinting);

вбудовування заголовків (captioning).

Стеганосистема виконує завдання вбудовування і виділення повідомлень з іншої інформації. Вона складається з наступних основних елементів, поданих на рис.1.1:

попередній кодер (перекодер) – призначений для перетворення прихованого повідомлення до виду який зручний для вбудовування в сигнал-контейнер. (Контейнером є інформаційна послідовність, в якій приховується повідомлення);

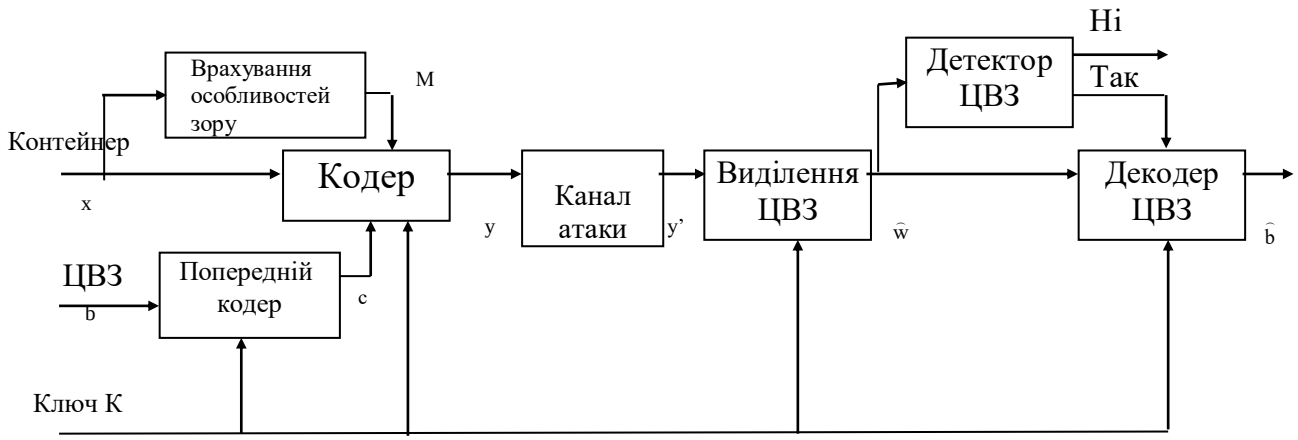


Рис.1.1. Структурная схема типичной стегосистемы ЦВЗ

стегокодер – пристрій, призначений для вкладення прихованого повідомлення в другі дані з врахуванням їх моделі;

облаштування виділення вбудованого повідомлення;

стегодетектор – пристрій, для визначення наявності стегоповідомлення;

декодер – пристрій який поновлює приховане повідомлення.

Дані, в яких знаходиться приховане повідомлення, можуть піддаватися умисним атакам або випадковим завадам.

На рис.1.1, в стеганосистемі об’єднується два типи інформації так, щоб вони могли бути помітні двома принципово різними детекторами. Одним виступає система виділення ЦВЗ, в якості іншого - людина.

На рис.1.2 наведена класифікація системи стеганографії.

Щоб стеганосистема була надійною, потрібне виконання при її проектуванні ряд вимог:

безпека системи визначається секретністю ключа. Це значить, що порушник знає алгоритми роботи стеганосистеми і статистичні характеристики великої кількості повідомлень і контейнерів, і це не дає йому додаткової інформації про повідомлення в цьому контейнері;

знання супротивником факту наявності повідомлення в любому із контейнерів не допоможе при виявленні повідомлень в інших контейнерах;

заповнений контейнер має бути візуально непомітний від порожнього. Для виконання вимоги необхідно впроваджувати повідомлення яке приховане у незначні області сигналу. Ці області використовують алгоритми стиснення.

Коли зображення піддаватиметься стисненню, тоді приховане повідомлення зруйнується;

повинна бути забезпечена потрібна пропускна здатність (вимога актуальна, в більшості, для стегосистем прихованої передачі);

також стеганосистема повинна мати прийнятну для обчислювальння складність реалізації. При цьому можлива несиметрична по складності реалізації система ЦВЗ, тобто простий стеганодекодер і складний стеганокодер;

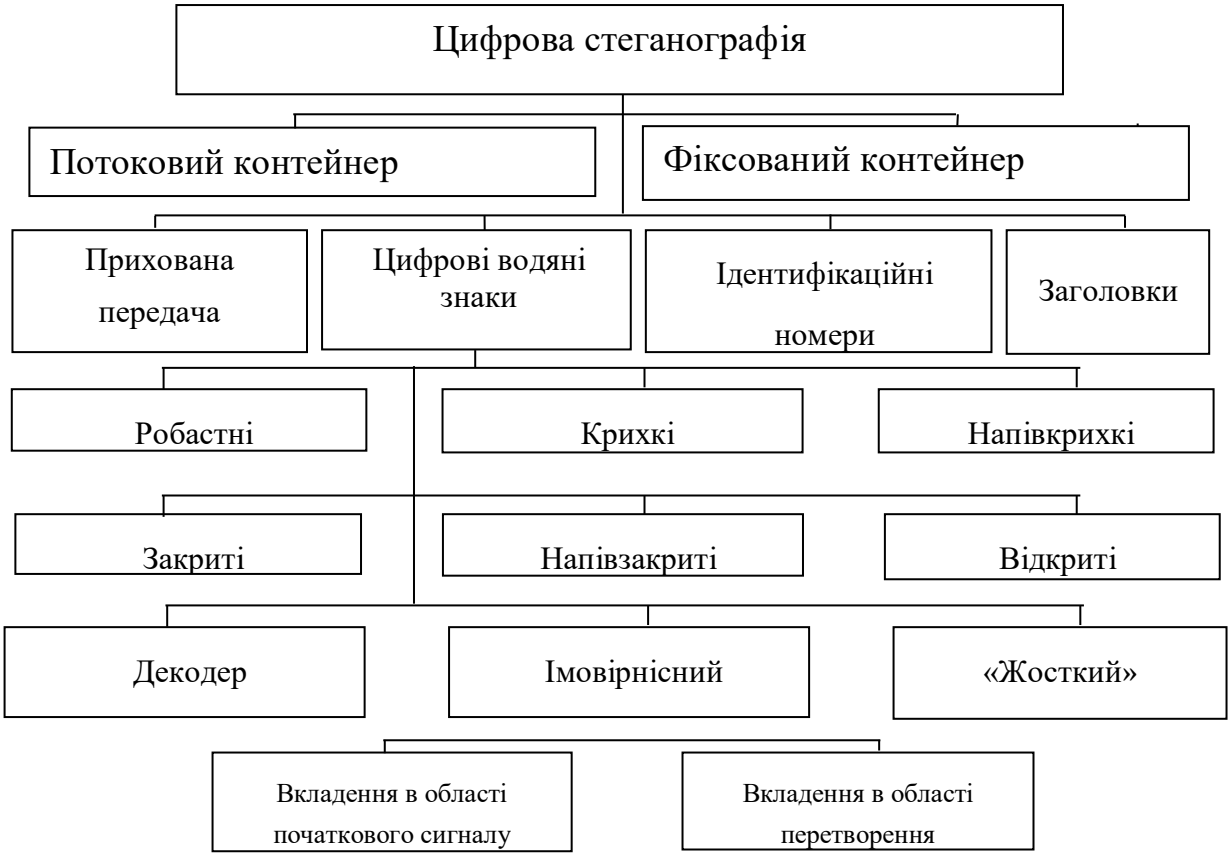


Рис.1.2. Класифікація системи цифрової стеганографії.

Цифрові зображення представляють з себе матрицю пікселів. Піксель – це одиничний елемент зображення. В нього фіксована розрядність двійкового представлення. Наприклад, пікселі півтонового зображення мають код 8 біт (значення яскравості від 0 до 255).

Молодший значущий біт (LSB) зображення має в собі найменше інформації. Відомо, що людина зазвичай не здатна помітити зміну в цьому біті. Фактично, він шум. Тому його використовують для вбудовування інформації. Таким чином, для півтонового зображення об’єм вбудовуваних даних може скласти 1/8 об’єму контейнера. Наприклад, в зображенні розміром 512x512

можна вбудувати 32 кілобайти інформації. Якщо модифікувати два молодші біти (що майже непомітно), то можна таємно передати вдвічі більший об’єм даних.

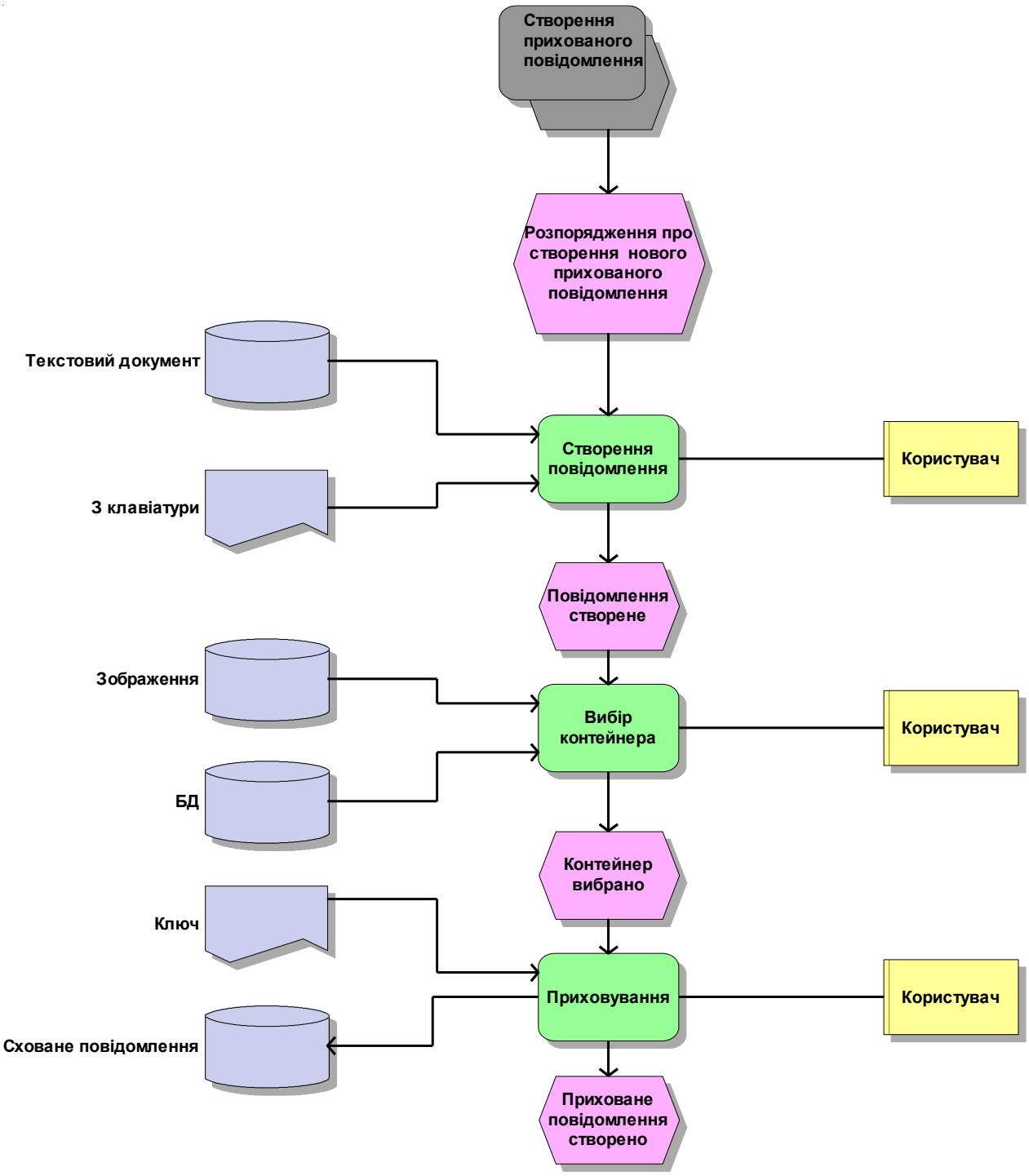


Рис 1.3. Схема бізнес процесу «Створення прихованого повідомлення»

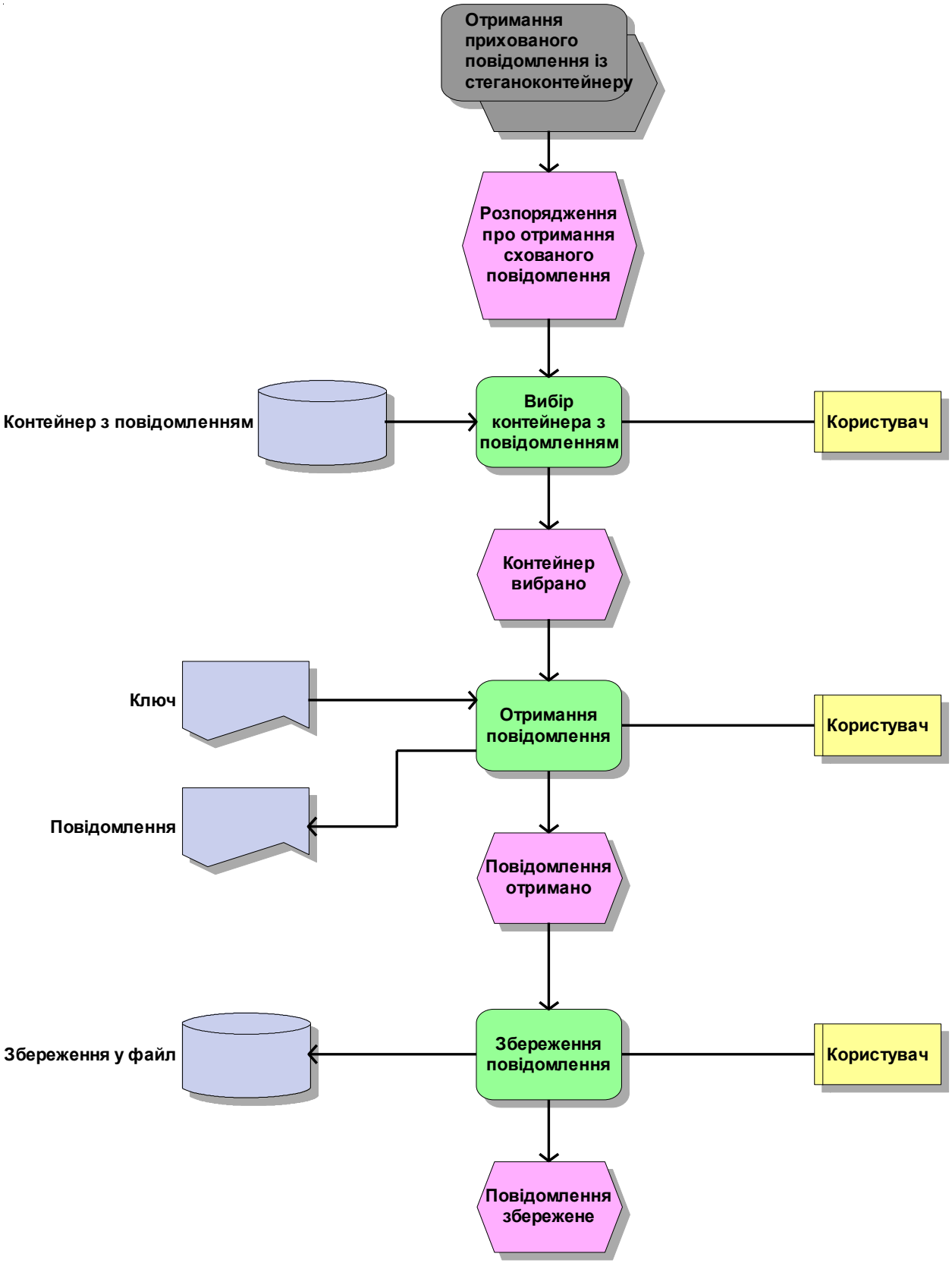


Рис 1.4. Схема бізнес процесу «Отримання схованого повідомлення із стеганоконтейнеру»

Перевага даного методу полягає в простоті і великому об’ємі вбудовуваних даних. Проте, є серйозні недоліки. Приховане повідомлення легко руйнується, як це показано в третій главі. Не забезпечується секретність вбудовування інформації. Порушникові точно відоме місце розташування усього ЦВЗ. Для усунення останнього недоліку запропоновано вбудовувати ЦВЗ не у всі пікселі зображення, а в деякі з них, котрі визначаються за псевдовипадковим законом відповідно ключу, відомому тільки авторизованому користувачеві. Пропускна здатність при цьому зменшується.

На рисунках 1.3 і 1.4 наведена схема бізнес процесу створення прихованого повідомлення, тобто вбудовування ЦВЗ в зображення і витягування його.

1.3. Аналіз існуючого програмного продукту, що реалізують функцію предметної області

ImageSpyer призначена для приховування будь-яких файлів в картинці, без її видимого змінення і спотворення.

Рис 1.5. Інтерфейс користувача утиліти ImageSpyer.

Програма використовує алгоритм який дозволяє приховати в картинці об’єм даних рівний кількості пікселів зображення, тобто приблизно 1/3 від

розміру файлу зображення. При цьому приховані дані абсолютно захищені, оскільки вони шифруються індивідуальним паролем користувача з використанням 40 стійких до дешифрування і перебору алгоритмів шифрування, серед яких AES, GOST, Blowfish, Twofish, IDEA, CAST 256, Skipjack, RC6 та ін. Окрім цього, розпізнати приховані дані складно за рахунок додаткових можливостей утиліти: ручна установка секретного порядку біт і ідентифікатора даних. Програма украй проста у використанні інтерфейс користувака приведено на рис 1.3, працює швидко і надійно.

Рис 1.6 Інтерфейс користувача Steganos Security Suite

Steganos Security Suite (раніше називалася Security Suite) найвідоміша з програм, використовуючих стеганографию. Зашифровані дані "ховаються" у файлах графіки (наприклад, малюнку), або музичному файлі, які зовні нічим не

відрізняються від аналогічних файлів, що не несуть зашифрованої інформації, - картинку можна подивитися, музику можна послухати.

Оцифровані файли (*.bmp або *.wav) змінюються певною мірою, і це не впливає на якість звуку або зображення (ці зміни будуть практично не помітні).

Від аналогічних програм Steganos Security Suite вигідно відрізняється здатністю приховувати дані не лише у файлах форматів wav і bmp, але і в html і навіть в звичайних текстових. Не буде зайвої і наявна функція видалення файлів без можливості їх відновити. Окрім цього, програма підтримує 256-бітове шифрування за стандартом AES, видалення усіх слідів знаходження в Інтернеті, має модуль шифрування пошти, що дозволяє вести шифроване листування, а також менеджер паролів. Також є опція блокування комп'ютера від сторонніх. Інтерфейс користувача приведено на рисунку 1.6.

Програмний пакет включає в себе декілька утиліт. File Manager –утиліта призначена для зашифрування вмісту файлів і папок, також для приховування їх від чужих очей. Якщо просто зашифрувати файли і папки, програма утворить файл з розширенням .sef, для його відкриття необхідно ввести пароль. При створенні цього файлу програма попросить вказати, чи необхідно видалити початкові файли з жорсткого диска, або ж їх необхідно зберегти.

Рис 1.9. Інтерфейс File Manager.

Але функція приховання даних, котра реалізована в File Manager, набагато цікавіша. Використанням її, не просто зашифровують файли, але і їх ховають в будь-якому нешкідливому графічному чи аудіофайлі. При цьому у файловому менеджерові відобразиться тільки файл-прикриття, а вміст

графічного файлу можна подивитися, при відкритті його в File Manager і введенні паролю. Ця функція зручна, бо файли з розширенням .sef здатні притягувати увагу, а нешкідливі зображення або файли MP3 - навряд чи.

Таблиця 1.1

Порівняльна характеристика ImageSpyer і Steganos Security Suite.

Фірма-розробник	Dark	Steganos
Назва програмного продукту	ImageSpyer	Steganos Security Suite
Версії продукту	1.0	11
Функціональність	підтримка приховання даних в пікселах TIFF файлу (LZW, ZIP, PIXAR компресія) підтримка усіх 40 криптоалгоритмів	програмний пакет включає в себе дев’ять інструментів для маскування інформації:
Швидкість роботи	середня	середня
Забезпечення конфіденційності	не забезпечує	забезпечує (суттєво знижує шводкість)
Вибір стеганоконтейнера	безальтернативний	безальтернативний
Опис	утиліта проста у використанні, і зрозуміла для будь-якого користувача	програмний пакет більш складний, має багато функцій для сховування інформації
Тощо	безкоштовна	Пробна версія 1 місяць
Офіційний сайт	http://amsoftware.narod.ru/	http://www.steganos.com

Система відноситься до систем з безальтернативним варіантом формування стеганоконтейнеру, тому суттєвим недоліком є непрозорість формування стеганоконтейнеру для користувача. Steganos Security передає інформацію по каналам зв’язку з вірогідністю 10^{-3}–10^{-2} (повітряні лінії зв’язку) помилки на прийомній стороні, розшифровка спотвореної інформації неможлива. Виходячи з цього стає актуальною задача розробки програмного продукту який усуває ці недоліки.

В табл. 1.1 наведена порівняльна характеристика ImageSpyer і Steganos Security Suite.

РОЗДІЛ 2

2.1. Глосарій проекту

Глосарій – це словник основних термінів, які використовуються. Документ є найперший результат концептуального аналізу предметної області. Глосарій може бути розглянутий як документ, що засвідчує розуміння основних термінів Замовником і Розробником.

Також, глосарій є фундаментом для побудови розгорнених моделей предметної області, котрі на стадії реалізації ІС лягають в основу моделі даних (для генерації схеми бази даних) та об’єктної моделі (для об’єктно-орієнтованих застосувань).

Глосарій проекту подано в табл. 2.1

Таблиця 2.1

Глосарій проекту

Термін	Опис терміну
1	2
1. Основні поняття і категорії предметної області та проекту	
Повідомлення	термін, що використовується для загальної назви передаваної та закритої інформації.
Цифрова стеганография	напрям класичної стеганографии, заснований на прихованні або впровадженні будь-якої додаткової інформації в цифрові об’єкти, що викликає при цьому спотворення цих об’єктів.
Стеганографія	це наука про приховання передачі інформації через збереження в таємниці факту передавання.
Стеганографическая система (стеганосистема)	об’єднання методів і засобів які використовуються для створення прихованого каналу для передавання інформації. При побудові даної системи враховувалося те, що : 1) зловмисник знає алгоритм роботи стеганографической системи. Невідомим для супротивника є ключ за яким можна дізнатися факт існування та зміст закритого повідомлення.

Продовження табл. 2.1

1	2
	2) При виявленні супротивником прихованого повідомлення він не зможе витягнути його поки не володітиме ключем. 3) Супротивник не має технічних і інших переваг.
Стеганокодер	пристрій, який призначається здійсювати вкладення прихованого повідомлення в інші дані з врахуванням їх моделі;
Декодер	пристрій, поновлюючий приховане повідомлення. Цей вузол може бути відсутнім, як буде пояснено далі.
Цифровий водяний знак (ЦВЗ)	використовують для забезпечення захисту від копіювання, та збереження авторських прав.
Метод найменш значущих бітів	суть цього методу полягає в заміні останніх значущих бітів контейнера (відеозаписи, аудіо або зображення) на біти повідомлення яке приховане. Різниця між пустим і повним контейнерами повинна бути не помітна для людських органів сприйняття.
Контейнер	так називають будь-яку інформацію, що використовується для приховання закритого повідомлення.
Порожній контейнер	котрий не містить секретного послання.
Заповнений контейнер (стегоконтейнер)	контейнер, який містить секретне послання.
Стеганографічний канал (стеганоканал)	передає стеганоконтейнер.
Ключ (стеганоключ)	необхідний для приховання стеганоконтейнера. Ключі в стеганосистемах є 2 типів: секретні і відкриті. Якщо використовується секретний ключ, він повинен створюватися до початку обміну повідомленнями, або передаватися по захищеним каналом. Стеганосистема, яка використовує відкритий ключ, повинна бути влаштована так, щоб неможливо отримати з закритого відкритий ключ.

Закінчення табл. 2.1

1	2
	Відкритий ключ можемо передавати по незахищеному каналу.
Стегодетектор	пристрій, призначений для визначення наявності стеганоповідомлення;
Коди БЧХ	Коди (Боуза Чоудхури Хоквінгхема) широкий клас циклічних кодів, застосовуваних для захисту інформації від помилок
Двійкова відносна фазова модуляція (BPSK)	найпростіша форма фазової маніпуляції. Алгоритм схеми двійковій ФМн полягає в зміщенні фази несучих коливань на одне з двох значень, нуль або π (180 °).
2. Користувачі системи	
Користувач	шифрує та відсилає зашифроване повідомлення
Криптоаналітик	фахівець з криптоаналізу
3. Вихідні та вхідні документи	
Таємна інформація	інформація яка містить відомості, що становлять державну та іншу таємницю передбачену законом, розголошення якої завдасть шкоди особі, суспільству та державі. Є різновидом інформації з обмеженим доступом.
Пустий та заповнений стеганоконтейнер	зображення яке містить, або не містить вбудоване повідомлення.

2.2. Розроблення варіанту використання

Мета варіантів використання в тому, щоб визначити закінчений аспект чи фрагмент поведінки будь-якої сутності не розкриваючи її внутрішню структуру. В якості цієї сутності може виступити початкова система або будь-який другий елемент моделі, що володіє власною поведінкою, що подібно цій підсистемі чи класу в моделі системи.

Варіант використання відповідає окремому сервісу, котрий надає модельовану сутність чи систему про запит користувача (актора), іншими словами визначає спосіб застосування цієї сутності. Сервіс, який інстанціюється на запит користувача, являє собою закінчену послідовність дій.

Це значить, що коли система закінчить обробляти запит користувача, вона змушена повернутися в вихіде положення до виконання наступних запитів.

Також варіанти використання допомагають:

сфокусуватися на зовнішній поведінці системи окремо від її внутрішньої будови;

описати потреби користувачів і зацікавлених осіб;

скласти узгоджений глосарій термінів для користувачів, розробників і тест-инженерів;

зменшити число пропусків і невідповідностей у вимогах;

спростити реагування на зміни вимог;

планувати послідовність розробки функцій;

використовувати моделі як основу для системних тестів, встановлюючи чітке відношення між тестами і вимогами. При зміні вимог це відношення допомагає правильно оновлювати тести. Це дозволяє забезпечити відповідність системи новим вимогам.

Схеми варіантів використання створюються, щоб описати, хто і для чого використовує систему. Варіант використання представляє мету користувача системи і процедуру, що виконується користувачем для досягнення цієї мети.

2.2.1. Розроблення діаграми варіантів використання.

Для більш точного зрозуміння як повинна працювати система, частіше використовується опис її функціональності через варіанти використання (Use Case чи прецеденти). Варіанти використання це - описання послідовності дій, котрі може здійснювати система в противагу на зовнішні впливи користувачів чи будь-яких інших програмних систем або системи. Варіант використання показує функціональність системи з точки зору отримання результату користувачем, тому вони точно ранжують функції за значимістю отриманого результату.

Варіанти використання призначені для визначення функціональних вимог до системи та керують всім процесом розробки. Такі основні види діяльності як аналіз, проектування, тестування в основному виконуються за допомогою варіантів використання. Аналіз і проектування варіантів використання дають розуміння того як результати, котрі хоче отримати користувач мають вплив на архітектуру системи і як мають себе вести компоненти системи, щоб реалізувати потрібну користувачеві функціональність.

Під час тестування, описані раніше варіанти використання простіше оцінюють точність реалізації вимог користувачів і проводять покрокову перевірку цих вимог.

Діаграма варіантів використання наведена на Рис. 2.1.

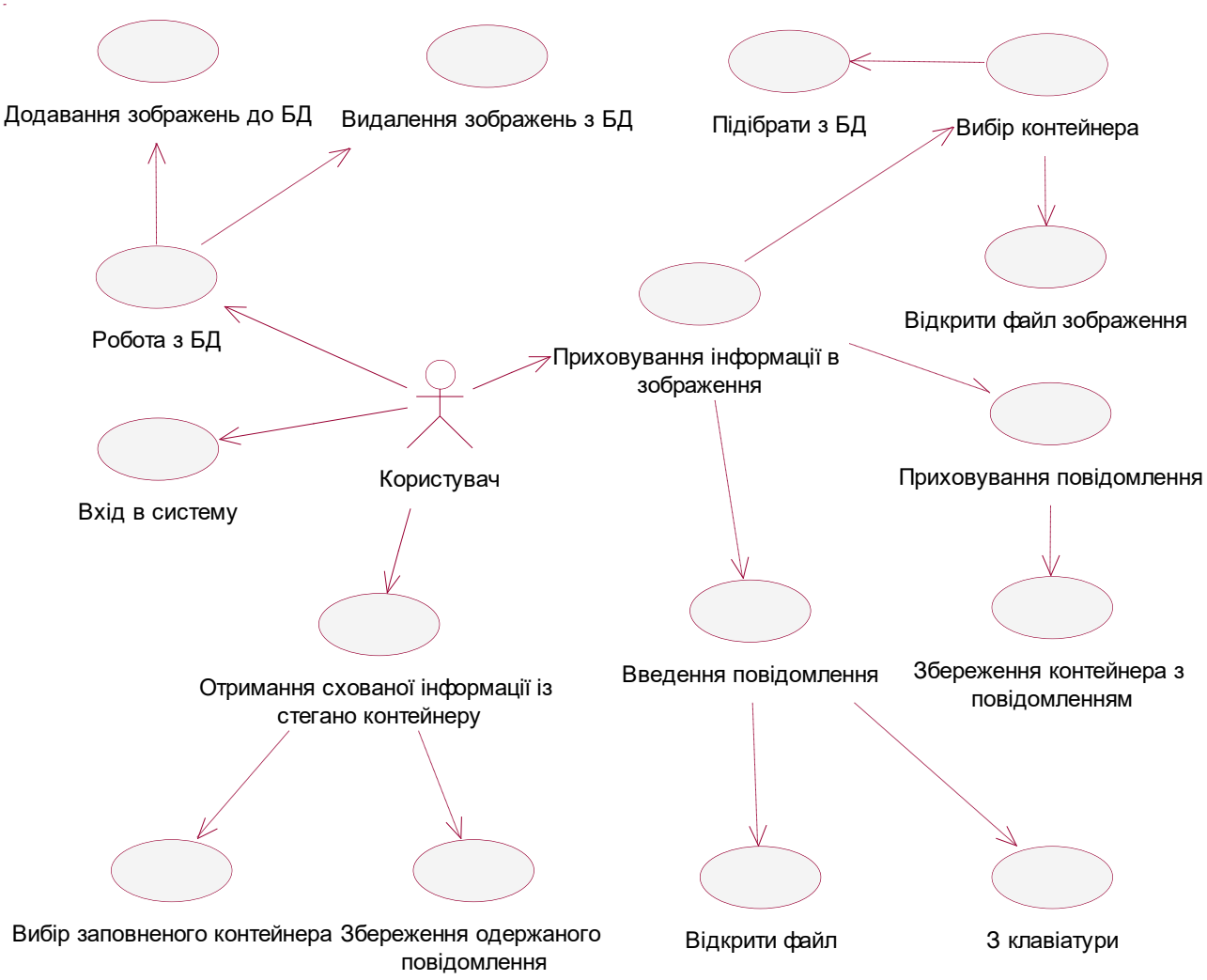


Рис. 2.1. Діаграма варіантів використання

2.2.2. Специфікація варіантів використання

Специфікація варіантів використання створюється на основі варіантів використання системи, описуючи такі характеристики варіанту використання як діюча особа, передумова, постумова, тригер, сценарій.

Варіант використання «Вхід в систему» табл. 2.1.

Таблиця 2.1

Варіант використання «Вхід в систему»

Контекст використання	Використовується при початку роботи з програмою
Діюча особа	Користувач
Передумова	Користувач повинен скрити інформацію
Тригер	Запуск програми
Сценарій	Користувач отримує розпорядження про створення нового зашифрованого повідомлення, запускає програмний додаток, вводить свій логін та пароль.
Постумова	Користувач аутентифікований в системі

Варіант використання «Приховування інформації в зображення» табл. 2.2.

Таблиця 2.2

Варіант використання «Приховування інформації в зображення»

Контекст використання	Використовується при необхідності сховати інформацію
Діюча особа	Користувач
Передумова	Користувач повинен скрити інформацію
Тригер	Запуск програми
Сценарій	Користувач отримує розпорядження про створення нового прихованого повідомлення, запускає програмний додаток, вводить свій логін та пароль, натискає кнопку «Сховати»
Постумова	Користувач аутентифікований в системі.

Варіант використання «Введення повідомлення» табл. 2.3.

Таблиця 2.3

Варіант використання «Введення повідомлення»

Контекст використання	Використовується для введення повідомлення
Діюча особа	Користувач
Передумова	Користувач повинен ввести секретне повідомлення
Тригер	Запуск програми
Сценарій	Користувач отримує розпорядження про створення нового зашифрованого повідомлення, запускає програмний додаток, вводить свій логін та пароль, вводить повідомлення.
Постумова	Користувач аутентифікований в системі.

Варіант використання «Відкрити файл» табл. 2.4.

Таблиця 2.4

Варіант використання «Відкрити файл»

Контекст використання	Використовується для введення повідомлення з файлу
Діюча особа	Користувач
Передумова	Користувач повинен ввести секретне повідомлення
Тригер	Запуск програми
Сценарій	Користувач отримує розпорядження про створення нового зашифрованого повідомлення, запускає програмний додаток, вводить свій логін та пароль, відкриває файл з секретною інформацією.
Постумова	Користувач аутентифікований в системі.

Варіант використання «Введення повідомлення з клавіатури» табл. 2.5

Таблиця 2.5

Варіант використання «Введення повідомлення з клавіатури»

Контекст використання	Використовується для введення повідомлення з клавіатури
Діюча особа	Користувач
Передумова	Користувач повинен ввести секретне повідомлення
Тригер	Запуск програми
Сценарій	Користувач отримує розпорядження про створення нового зашифрованого повідомлення, запускає програмний додаток, вводить свій логін та пароль, вводить повідомлення з клавіатури.
Постумова	Користувач аутентифікований в системі.

Варіант використання «Приховування повідомлення» табл. 2.6

Таблиця 2.6

Варіант використання «Приховування повідомлення»

Контекст використання	Використовується для приховування повідомлення
Діюча особа	Користувач
Передумова	Користувач повинен приховати секретне повідомлення
Тригер	Запуск програми
Сценарій	Користувач отримує розпорядження про створення нового зашифрованого повідомлення, запускає програмний додаток, вводить свій логін та пароль, вводить повідомлення, натискає «Приховати».
Постумова	Користувач аутентифікований в системі, вибрав контейнер і ввів секретне повідомлення.

Варіант використання «Збереження контейнера з повідомленням» табл. 2.7

Таблиця 2.7

Варіант використання «Збереження контейнера з повідомленням»

Контекст використання	Використовується для збереження контейнера з повідомленням
Діюча особа	Користувач
Передумова	Користувач повинен зберегти сховане секретне повідомлення
Тригер	Запуск програми
Сценарій	Користувач отримує розпорядження про створення нового зашифрованого повідомлення, запускає програмний додаток, вводить свій логін та пароль, вводить повідомлення, натискає «Зберегти».
Постумова	Користувач сховав секретне повідомлення.

Варіант використання «Вибір контейнера» табл. 2.8.

Таблиця 2.8

Варіант використання «Вибір контейнера»

Контекст використання	Використовується для вибору контейнера
Діюча особа	Користувач
Передумова	Користувач повинен вибрати контейнер
Тригер	Запуск програми
Сценарій	Користувач отримує розпорядження про створення нового зашифрованого повідомлення, запускає програмний додаток, вводить свій логін та пароль, натискає «відкрити зображення».
Постумова	Користувач ввів секретне повідомлення.

Варіант використання «Підібрати з БД» табл. 2.9.

Таблиця 2.9

Варіант використання «Підібрати з БД»

Контекст використання	Використовується для вибору контейнера з БД
Діюча особа	Користувач
Передумова	Користувач повинен вибрати контейнер
Тригер	Запуск програми
Сценарій	Користувач отримує розпорядження про створення нового зашифрованого повідомлення, запускає програмний додаток, вводить свій логін та пароль, натискає «Підібрати з БД».
Постумова	Користувач ввів секретне повідомлення.

Варіант використання «Вибір контейнера з файлу» табл. 2.10.

Таблиця 2.10

Варіант використання «Вибір контейнера з файлу»

Контекст використання	Використовується для вибору контейнера з файлу
Діюча особа	Користувач
Передумова	Користувач повинен вибрати контейнер
Тригер	Запуск програми
Сценарій	Користувач отримує розпорядження про створення нового зашифрованого повідомлення, запускає програмний додаток, вводить свій логін та пароль, натискає «відкрити зображення».
Постумова	Користувач ввів секретне повідомлення.

Варіант використання «Вилучення прихованого повідомлення» табл. 2.11.

Таблиця 2.11

Варіант використання «Вилучення прихованого повідомлення»

Контекст використання	Використовується для вилучення прихованого повідомлення
Діюча особа	Користувач
Передумова	Користувач повинен прочитати приховане повідомлення
Тригер	Запуск програми
Сценарій	Користувачу потрібно відкрити сховане повідомлення для цього він запускає програмний додаток, вводить свій логін та пароль.
Постумова	Користувач аутентифікований в системі.

Варіант використання «Вибір заповненого контейнеу» табл. 2.12.

Таблиця 2.12

Варіант використання «Вибір заповненого контейнеу»

Контекст використання	Використовується для вибору заповненого контейнеру
Діюча особа	Користувач
Передумова	Користувач повинен обрати контейнер з прихованим повідомленням
Тригер	Запуск програми
Сценарій	Користувачу потрібно відкрити сховане повідомлення для цього він запускає програмний додаток, вводить свій логін та пароль, обирає контейнер.
Постумова	Користувач аутентифікований в системі.

Варіант використання «Збереження одержаного повідомлення» табл. 2.13.

Таблиця 2.13

Варіант використання «Збереження одержаного повідомлення»

Контекст використання	Використовується для збереження повідомлення
Діюча особа	Користувач
Передумова	Користувач повинен зберегти приховане повідомлення
Тригер	Запуск програми
Сценарій	Користувачу потрібно відкрити сховане повідомлення для цього він запускає програмний додаток, вводить свій логін та пароль, обирає контейнер, натискає кнопку «Витягнути», натискає кнопку «Зберегти».
Постумова	Користувач аутентифікований в системі.

Варіант використання «Робота з БД» табл. 2.14.

Таблиця 2.14

Варіант використання «Робота з БД»

Контекст використання	Використовується для додавання та видалення контейнерів з БД
Діюча особа	Користувач
Передумова	Користувач повинен додати або видалити зображення з БД
Тригер	Запуск програми
Сценарій	Користувачу потрібно додати або видалити зображення з БД, для цього він запускає програмний додаток, вводить свій логін та пароль, натискає кнопку «Імпортувати зображення до БД».
Постумова	Користувач аутентифікований в системі.

Варіант використання «Додавання зображення до БД» табл. 2.15.

Таблиця 2.15

Варіант використання «Додавання зображення до БД»

Контекст використання	Використовується для додавання контейнерів в БД
Діюча особа	Користувач
Передумова	Користувач повинен додати зображення до БД
Тригер	Запуск програми
Сценарій	Користувачу потрібно додати або видалити зображення з БД, для цього він запускає програмний додаток, вводить свій логін та пароль, натискає кнопку «Імпортувати зображення до БД».
Постумова	Користувач аутентифікований в системі.

Варіант використання «Видалення зображень з БД» табл. 2.16.

Таблиця 2.16

Варіант використання «Видалення зображень з БД»

Контекст використання	Використовується для видалення контейнерів з БД
Діюча особа	Користувач
Передумова	Користувач повинен видалити зображення з БД
Тригер	Запуск програми
Сценарій	Користувачу потрібно додати або видалити зображення з БД, для цього він запускає програмний додаток, вводить логін та пароль, нажимає «Імпортувати зображення до БД».
Постумова	Користувач аутентифікований в системі.

2.2.3. Розкадровка варіантів використання

Розкадрування – це логічне та концептуальне описання функціональних можливостей системи під певний сценарій, включаючи потрібну взаємодію між системою і її користувачами.

Розкадровка використовується при проектуванні інтерфейсу користувача для ілюстрації знайдених рішень проектувальника інтерфейсу, та для отримання рекомендацій по інтерфейсу. Також вона надає можливість описувати свої потреби аналітикам, які, визначають вимоги до системи, здійснюють її перевірку відповідності до вимог поставленим завданням і здійснюють зворотний зв’язок з ними.

Переваги розкадровки:

а) видно функціональне призначення кожного елементу розкадровки;

б) наявність схеми навігації по програмному додатку;

в) дозволяє перевірити точність і повноту схеми;

г) може бути наведена користувачам для оцінки;

Розкадровка варіанту використання «Вхід в систему»

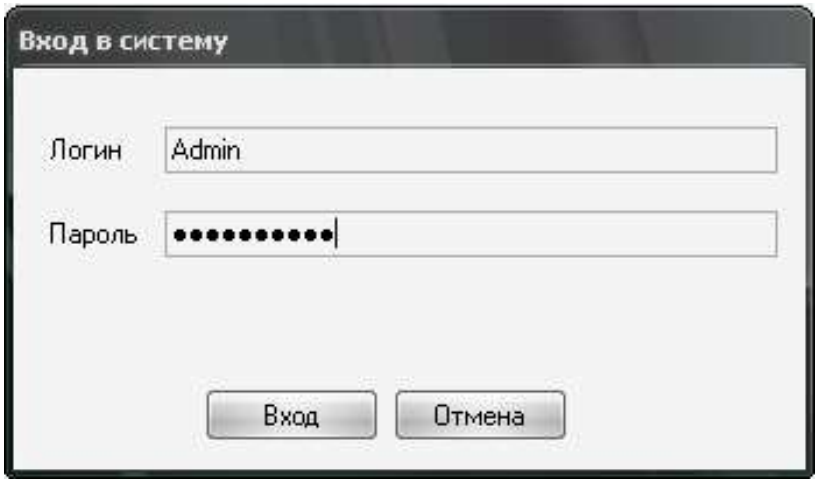


Рис.2.1. Вікно входу в систему

Після запуску додатку, користувач побачить вікно для авторизації. Коли користувач введе правильні логін та пароль буде показана головна форма.

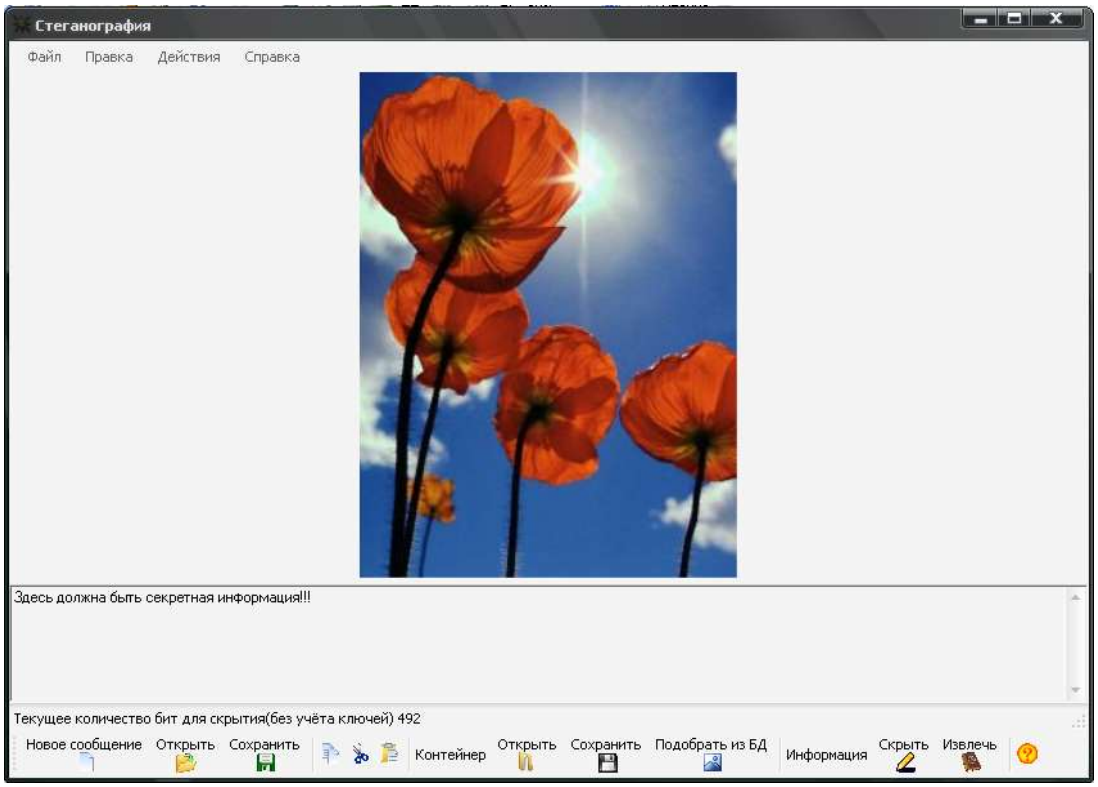


Рис.2.2. Головна форма додатку

Розкадровка варіанту використання «Сховування інформації в зображення»

Для сховування інформації потрібно натиснути «Скрыть» рис 2.2 і ввести ключі рис 2.3.

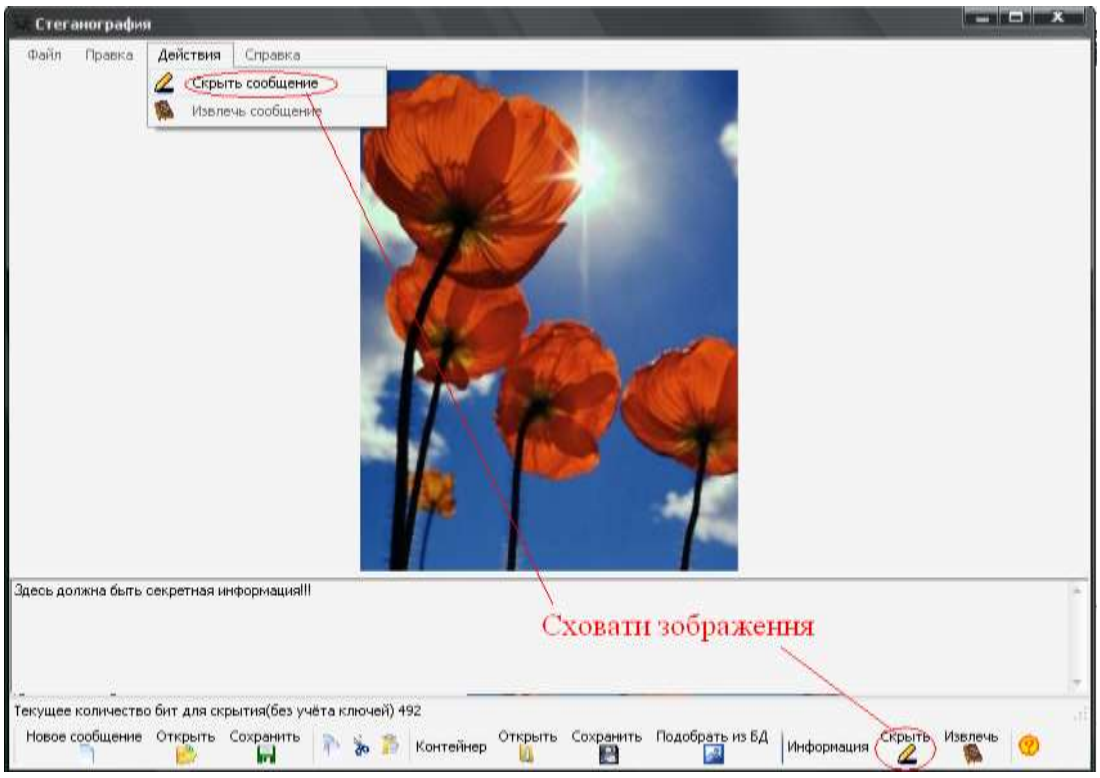


Рис 2.2 Сховати зображення

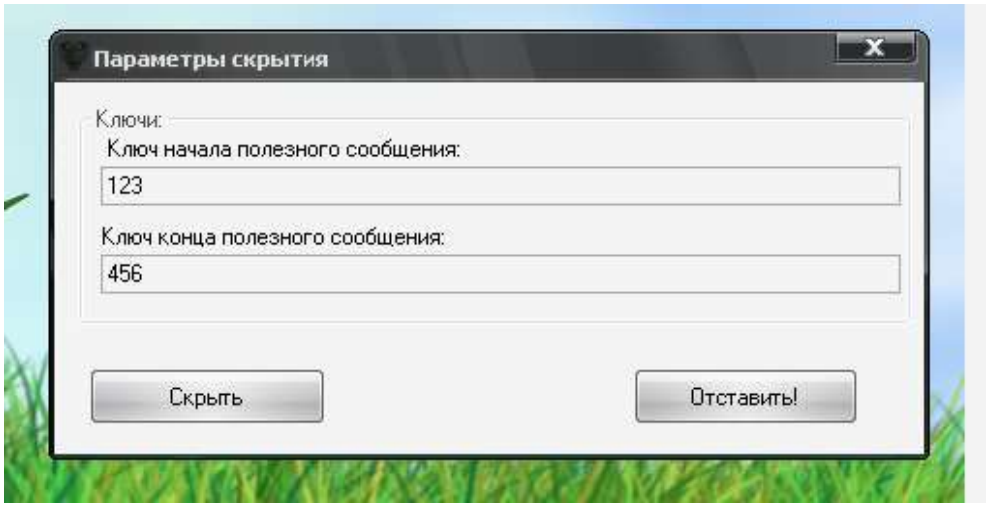


Рис 2.3. Сховування інформації в зображення.

Розкадровка варіанту використання «Введення повідомлення»

Щоб ввести повідомлення вам потрібно створити якесь повідомлення яке ви будете приховувати. Це можна зробити як вручну набравши текст повідомлення на клавіатурі, так і завантаживши його з файлу за допомогою головного меню(рис. 2.4).

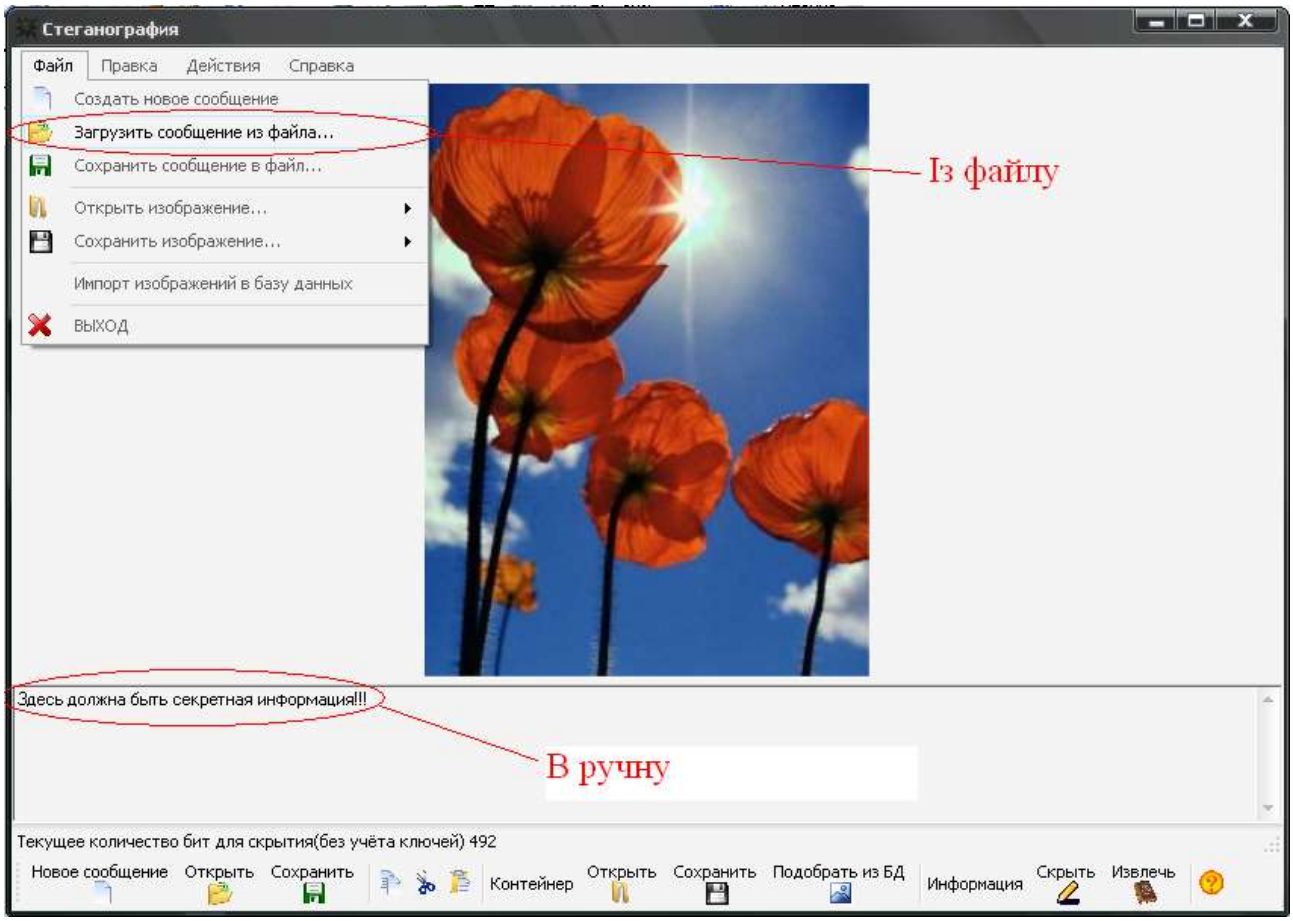


Рис 2.4. Введення повідомлення.

Розкадровка варіанту використання «Вибір контейнера в ручну».

Щоб відкрити контейнер вручну потрібно в нижньому меню натиснути на кнопку «Открыть» рис 2.5 і вибрати потрібні файли.

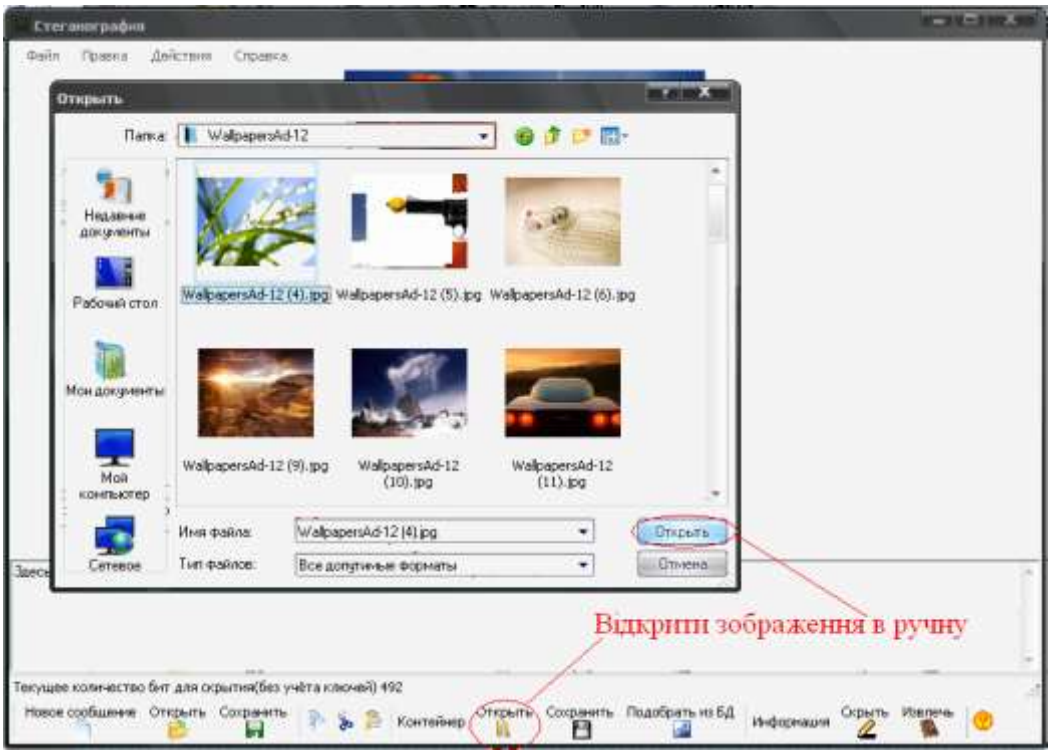


Рис 2.5. Вибір контейнера вручну.

Розкадровка варіанту використання «Вибір контейнера з БД».

Щоб підібрати контейнер з БД потрібно натиснути кнопку «Подобрать из БД», або в меню «Файл» вибрати «Открыть изображение из БД» рис 2.6. після цього відкриються доступні зображення рис 2.7.

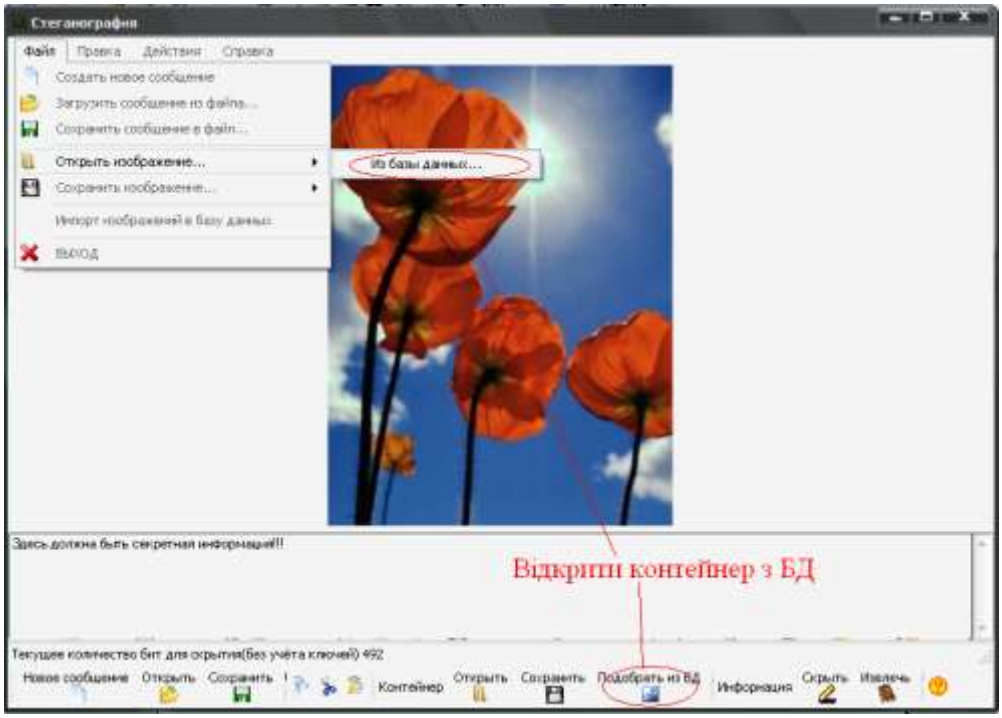


Рис 2.6. Вибір контейнера з БД .

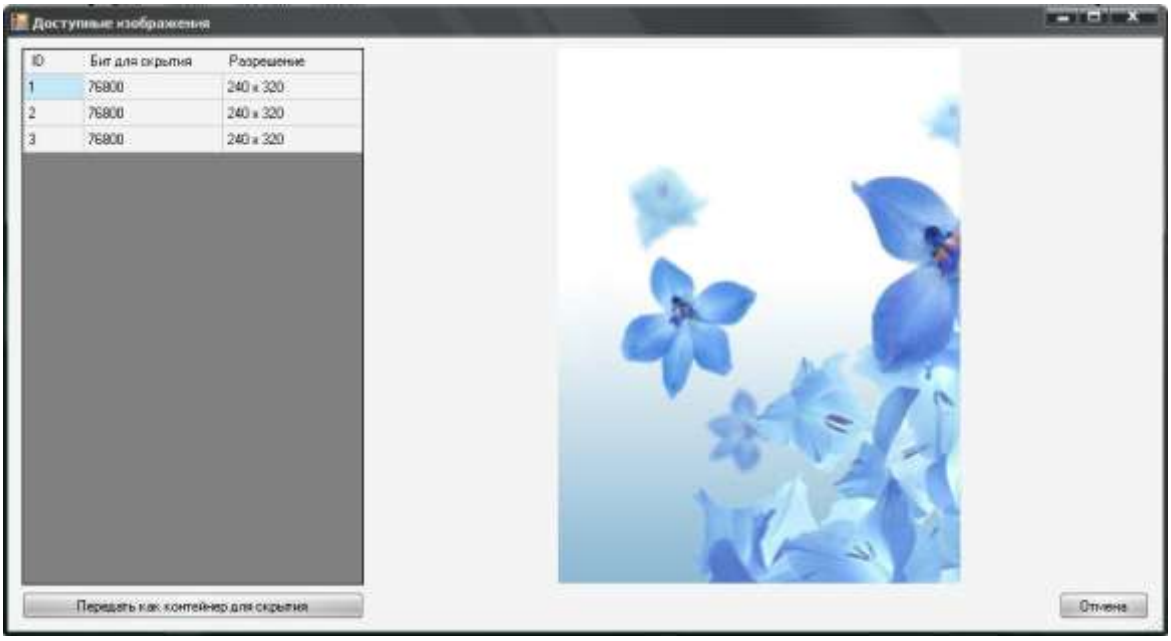


Рис 2.7. Доступні зображення в БД .

Розкадровка варіанту використання «Витягання схованої інформації».

Щоб витягнути інформацію потрібно натиснути в меню «Действия», «Извлечь сообщение» рис 2.8, та ввести ключі рис 2.9.

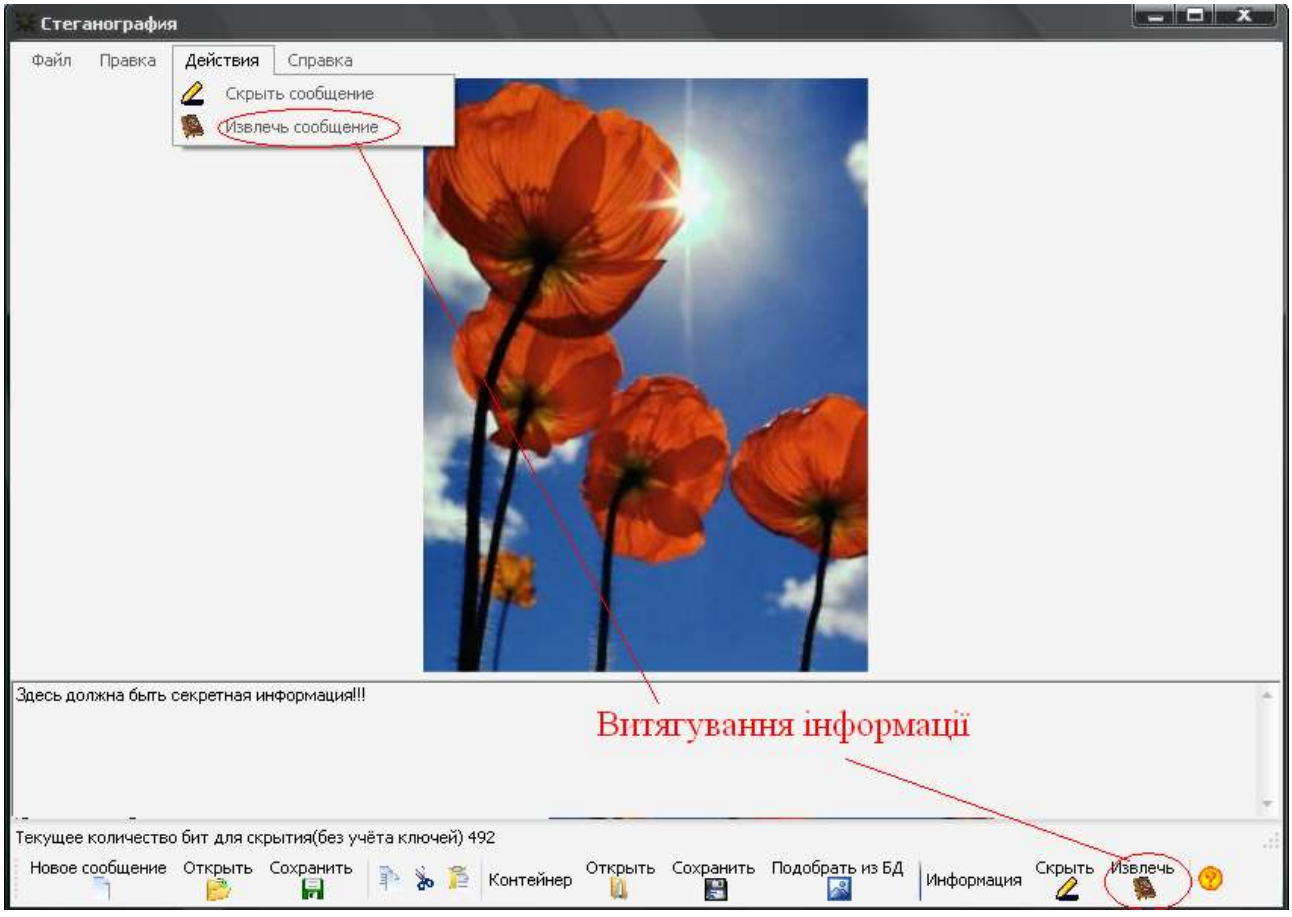


Рис 2.8. Витягання схованої інформації.

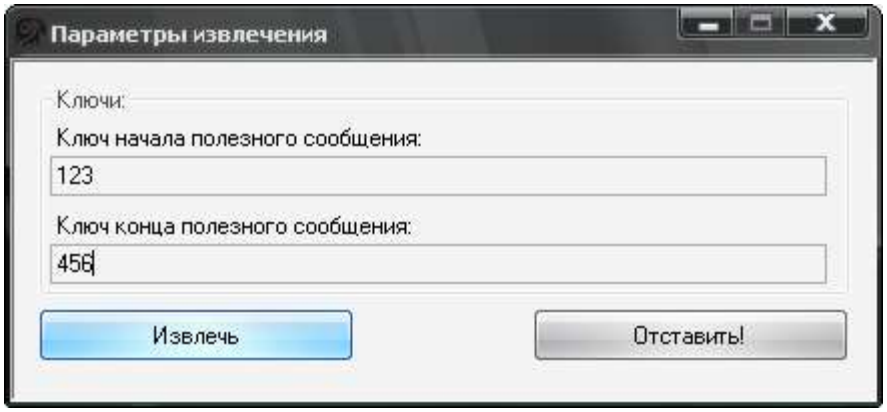


Рис 2.9. Параметри витягання.

Розкадровка варіанту використання «Збереження схованого повідомлення».

Щоб зберегти контейнер з прихованим повідомленням треба натснути кнопку «Сохранить» рис 2.10.

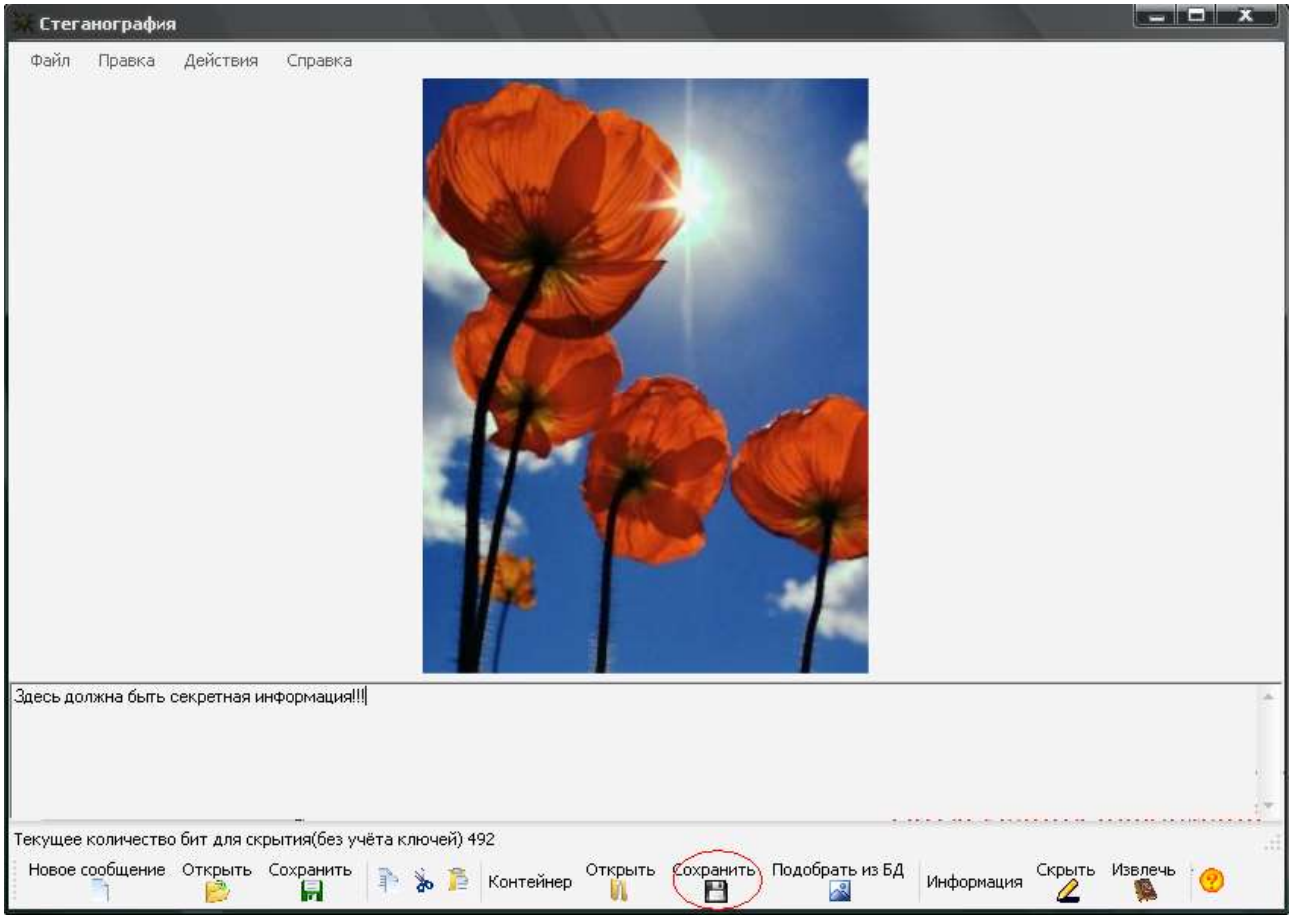


Рис 2.10. Збереження схованого повідомлення.

Розкадровка варіанту використання «Робота з БД».

Щоб додати зображення до БД потрібно відкрити меню «Файл» і вибрати «Импорт изображений в БД» рис 2.11. Обрати одне або декілька зображень і

натиснути «Открыть» рис 2.12. Кнопкою «Добавить вбраное» додати потрібні зображення рис 2.13.

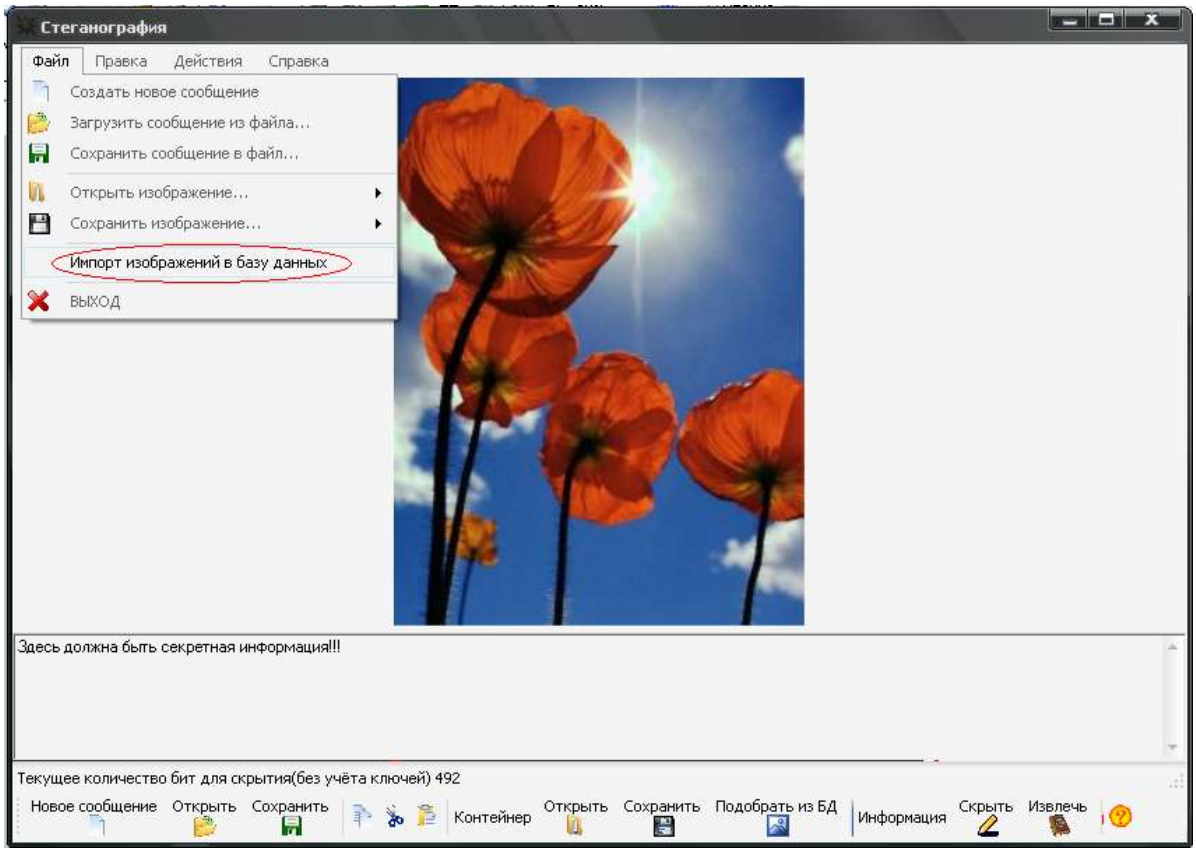


Рис 2.11.Імпорт до БД.

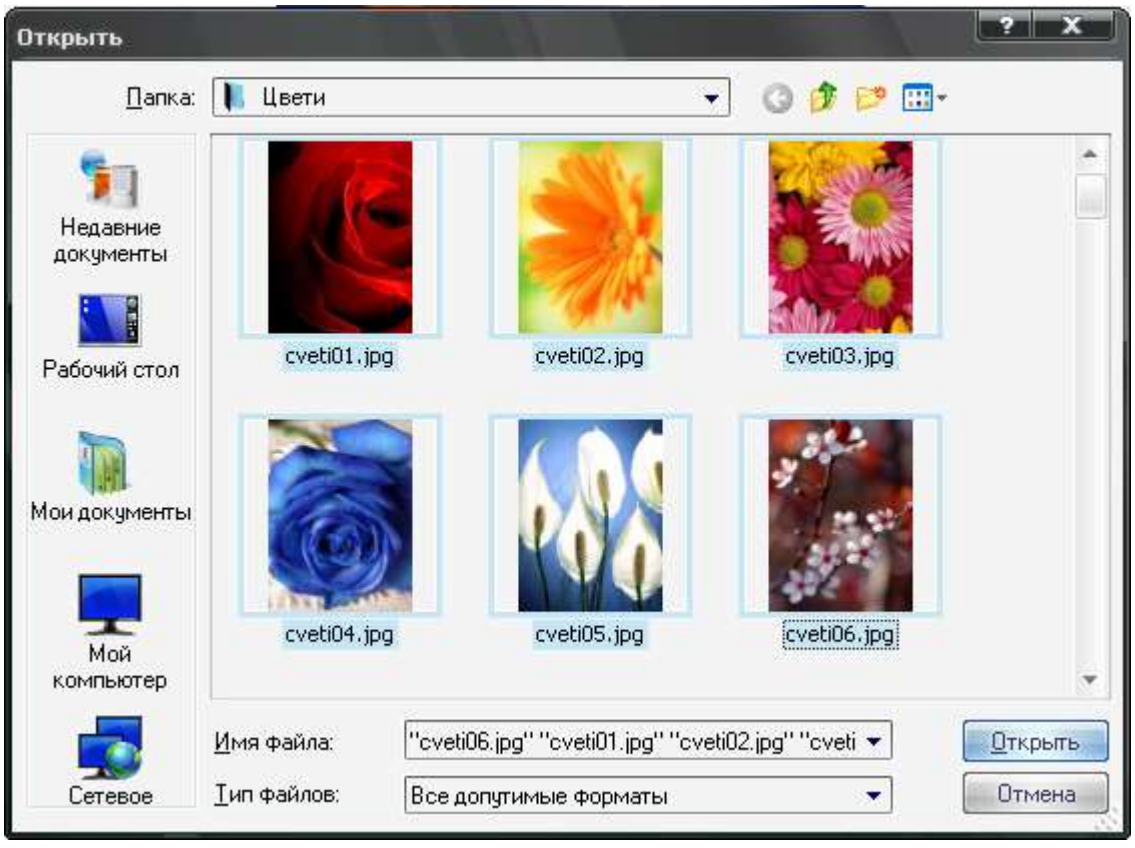


Рис 2.12. Додавання зображень.

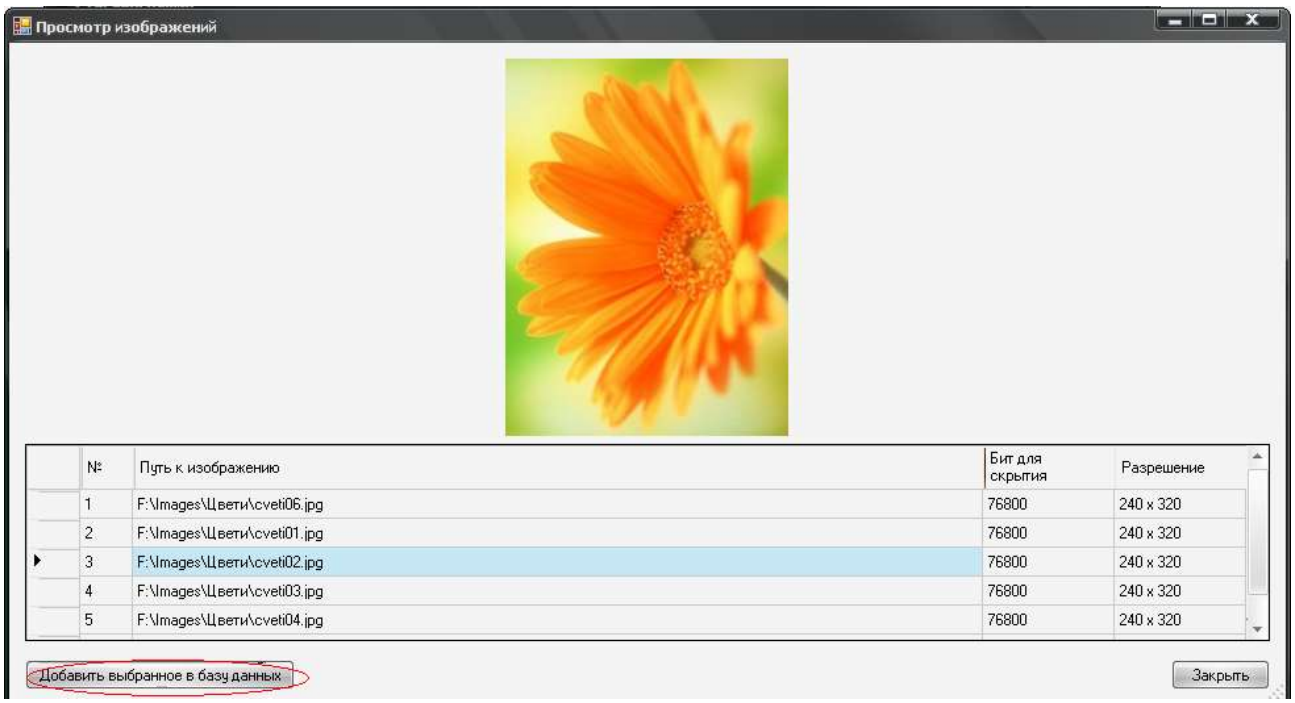


Рис 2.13. Перегляд зображень.

Розкадровка варіанту використання «Видалення зображень з БД».

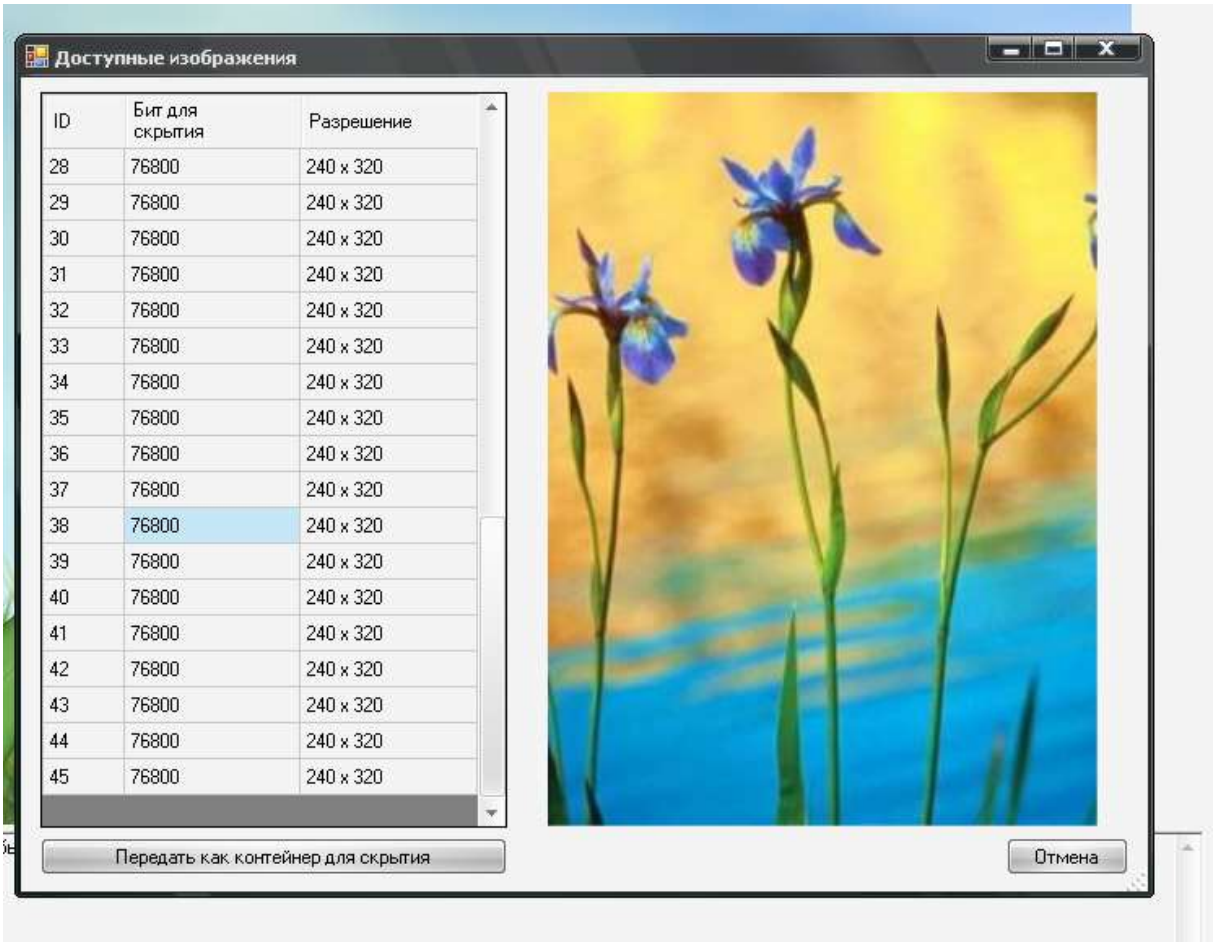


Рис 2.14. Видалення зображень з БД.

2.3. Специфікація функціональних і не функціональних вимог

Специфікація вимог, що стосуються програмного забезпечення (SRS) - це повний опис поведінки системи яка розробляється. До нього входить набір прецедентів, які описують взаємодію користувача з системою. Прецеденти відомі як функціональні вимоги. SRS також містить нефункціональні (чи додаткові вимоги). Нефункціональні вимоги являються вимогами котрі накладають обмеження на проєкт, або реалізацію (вимоги інженерії продуктивності, стандарти якості, та обмеження проектування).

Таблиця 2.1

Специфіка функціональних вимог

Ідентифікатор вимоги	Назва вимоги	Атрибути вимог		
		Пріоритети	Контакти	Труднощі
1	2	3	4	5
FR-UC-01	Вхід в систему	1	Користувач	середня
FR-UC-02	Отримання схованої інформації	1	Користувач	середня
FR-UC-03	Вибір заповненого контейнеру	2	Користувач	середня
FR-UC-04	Збереження схованого повідомлення	2	Користувач	середня
FR-UC-05	Сховування інформації в зображення	1	Користувач	середня
FR-UC-06	Введення повідомлення	2	Користувач	середня
FR-UC-07	Збереження контейнера з повідомленням	2	Користувач	середня
FR-UC-08	Вибір контейнера	2	Користувач	висока
FR-UC-09	Додавання зображення до БД	3	Користувач	середня
FR-UC-010	Видалення зображення з БД	3	Користувач	середня

Таблиця 2.14

Нефункціональні вимоги

Идентифи катор	Назва	Атрибути вимог		
		Пріоритети	Контакти	Труднощі
1	2	3	4	5
1. Застосовність				
SUPP-01	Час запуску системи – менше 5 сек.	1	Оператор	Середня
SUPP-02	Зручний і функціональний інтерфейс.	3	Оператор	Середня
SUPP-03	Легкість обслуговування системи (має бути передбачений відповідний функціонал для налагодження системи)	1	Оператор	Висока
2. Надійність				
SUPP-04	Мала кількість збоїв у роботі системи(2-3 у тиждень)	2	Оператор	Висока
SUPP-05	Стійка до збоїв, є можливість продовження роботи з системою у випадку збою	2	Оператор	Висока
3. Робоча характеристика				
SUPP-06	Час обробки – не більше 3 сек.	1	Оператор	Висока
4. Експлуатаційна придатність				
SUPP-07	Взаємодія системи з зовнішніми компонентами.	1	Оператор	Висока
5. Проектні обмеження				
SUPP-8	Мова програмування C#	1	Оператор	Низька
SUPP-9	СУБД - SQL SERVER	1	Оператор	Низька
6. Вимоги до документації і до системи допомоги призначеної для користувача				
SUPP-10	Повідомлення з попередженням про помилки виводяться у відповідних вікнах.	4	Оператор	Низька

Продовження табл. 2.14

1	2	3	4	5
SUPP-11	Наявність довідки користувача	4	Оператор	Низька
7. Куповані компоненти				
SUPP-12	Сумісність з OS Windows	4	Оператор	Низька
8. Інтерфейси				
8.1. Інтерфейси користувача				
SUPP-13	Єдине оформлення усіх вікон і повідомлень системи	1	Оператор	Низька
SUPP-14	Робота з БД здійснюється шляхом редагування, додавання, видалення записів з таблиць БД.	1	Оператор	Середня
8.2. Апаратні інтерфейси				
SUPP-15	512 Мбайт ОЗУ і вище	1	Оператор	Низька
SUPP-16	30 Мбайт вільного дискового простору;	1	Оператор	Низька
SUPP-17	Відеокарта з підтрианням роздільної здатності не менше 800 х 600 і можливості відображення більше 256 кольорів;	1	Оператор	Низька
SUPP-18	Монітор SVGA;	1	Оператор	Низька
SUPP-19	Процесор 800 МГц, або вище.	1	Оператор	Низька
8.3. Програмні інтерфейси				
SUPP-20	Наявність платформи .NET Framework 3.5 і вище	1	Оператор	Середня
SUPP-21	Наявність операційної системи Windows	1	Оператор	Середня
8.4. Комунікаційні інтерфейси				
SUPP-22	Протокол обміну муж клієнтами і сервером здійснено шляхом підключення до серверу БД – TCP.			

Закінчення табл. 2.14

1	2	3	4	5
9. Вимоги до ліцензування				
SUPP-23	Використання одної ліцензії на декілька робочих місць	1	Оператор	Середня
10.Застереження питань, пов’язаними з авторськими правами				
SUPP-24	Авторські права захищені законом	1	Оператор	Середня
11.Вживані стандарти				
SUPP-25	Стандарт якості програмного продукту ISO9001	1	Оператор	Середня

РОЗДІЛ 3

3.1 Математична постановка задачі

Стеганосистема може розглядатися як система зв'язку [8]. Алгоритм вбудовування ЦВЗ складається з наступних основних етапів: 1) генерування ЦВЗ, 2) вбудовування в кодері ЦВЗ, 3) виявлення ЦВЗ в детекторі.

1) Нехай *W*, K*, I*, B** є множина ЦВЗ, ключів, контейнерів і прихованих повідомлень. Тоді генерація ЦВЗ наведена у вигляді

$$F : I^* \times K^* \times B^* \to W^*, \quad W=F(I,K,B), \tag{1.2}$$

Власне кажучи, функція є довільна, але на практиці вимоги ЦВЗ накладають на неї деякі обмеження. У більшості випадків $F(I,K,B) \approx F(I+\varepsilon,K,B)$, змінений контейнер не змінює ЦВЗ. Функція *F* є складеною:

$$F = T \circ G \text{, де } G : K^* \times B^* \to C^* \text{ та } T : C^* \times I^* \to W^* \tag{1.3}$$

ЦВЗ залежить від властивостей контейнера, як це вже обговорювалося раніше. Функція G реалізовується за допомогою криптографічного надійного генератора ПСП з K в якості початкової величини.

Для підвищення стійкості ЦВЗ використовують завадостійкі коди, наприклад, коди БЧХ, згортальні коди [9]. В літературі показано позитивні результати вбудовування ЦВЗ в області інтегрального перетворення з використанням турбо-кодів. Значення ЦВЗ отримують значення {–1,1}, при цьому для відображення $\{0,1\} \to \{-1,1\}$ використовується двійкова фазова модуляція (BPSK).

Оператор T видозмінює кодові слова C*, в результаті отримано ЦВЗ W*. На функцію не накладається обмеження безповоротності, відповідний вибір G гарантує безповоротність F. Функцію T необхідно вибрати, щоб незаповнений контейнер I_0, заповнений I_w і модифікований заповнений I'_w породжували б аналогічний ЦВЗ:

$$T(C, I_0) = T(C, I_w) = T(C, I'_w), \tag{1.4}$$

тому має бути стійкість до невеликих змін самого контейнера.

2) Вбудовування ЦВЗ $W(i,j)$ у початкове зображення $I_0(i,j)$ описується як суперпозиція двох сигналів:

$$\varepsilon: \; I^* \times W^* \times L^* \to I_W^*, \quad I_w(i,j) = I_0(i,j) \oplus L(i,j) W(i,j) p(i,j), \tag{1.5}$$

де *L(I,j)* – маска вбудовування ЦВЗ, яка врахує зорову систему людини, використовується для зменшення видимості ЦВЗ;

p(i,j) – функція проектування, пов'язана з ключем;

знаком $\oplus$ оператор суперпозиції, що включає, окрім додавання, усічення та квантування.

Проектуюча функція забезпечує "розподіл" ЦВЗ з галузі картинки. Використовуючи її реалізовують рознесення в паралельних каналах інформації.

Можливість описати процес реалізації, представивши стеганосистему через систему зв'язку з передаванням додаткової інформації (рис.1.4)[8]. Модель в якій кодер і декодер крім ключа мають доступ до інформації про канал. В залежності від положення перемикача А і Б можна виділити чотири наступні класи стеганосистем (ключ весь час відомий кодеру і декодеру).

І клас: відсутня додаткова інформація (вимикачі розімкнуті) - "класична" стеганосистема. У ранніх джерелах по стеганографії вважається, що недоступна кодеру інформація про канал. Шляхом обчислення коефіцієнта кореляції здійснювалося виявлення ЦВЗ між прийнятим стего і очисленбим по ключу ЦВЗ.

Коли коефіцієнт перевищував поріг, надавалася ухвала про присутність ЦВЗ. Кореляційний приймач оптимальний в разі перешкоди Гауса. При других атаках стеганосистеми давали невтішні результати.

ІІ клас: тільки кодеру відома інформація про канал відома (А замкнуто, Б розімкнуто). Особливістю схеми є те, що, вона має ту ж пропускну можливість, що і схема з початковим контейнером в декодері. Недоліки стегосистеми ІІ класу є висока складність кодера (необхідно будувати кодову книгу для кожної картинки), та також відсутність адаптації прогнозованих атак. Запропоновані практичні підходи подолання цих недоліків. Для зменшення складності кодера використовують структуровані кодові книги, а на випадок найгіршої атаки декодер розраховують.

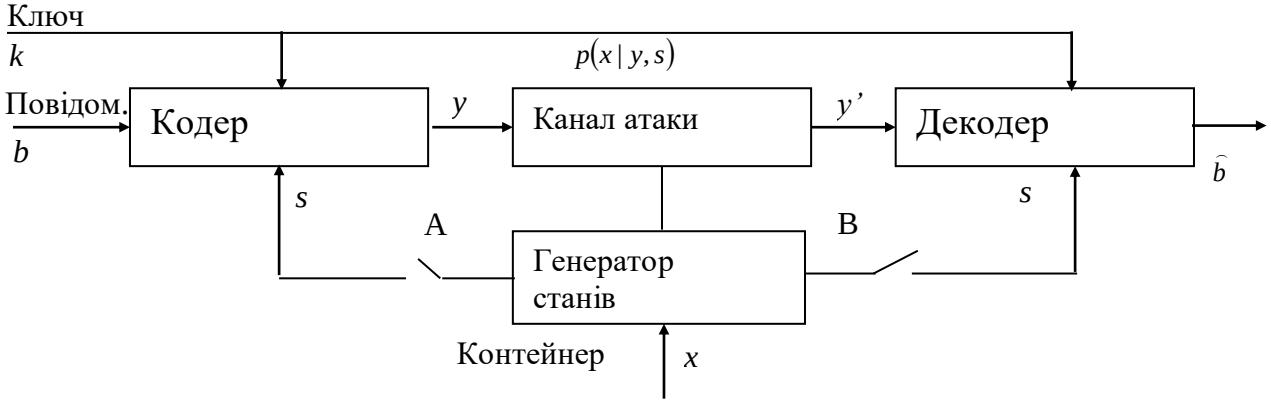


Рис.1.4. Представлення стегосистемы, як системи зв'язку з передачею додаткової інформації

III клас: декодеру відома тільки додаткова інформація (А розімкнуто, Б замкнуто). Декодер будується в даному випадку з врахуванням можливих атак. В наслідок чого виходять стійкі до геометричних атак системи. Методом досягнення мети є застосування опорного ЦВЗ (пілот-сигнал в радіозв'язку). Опорний ЦВЗ – мале число біт, вбудовані в незмінювані до перетворень коефіцієнти сигналу. Виконується вбудовування в амплітудні коефіцієнти перетворення Фур'є, як приклад, які незмінні до афінних перетворень. Після чог опорний ЦВЗ "вкаже", яке перетворення виконано із таємної інформації що атакує. Інша функція пілотного ЦВЗ боротьба із завмираннями, по аналогії з радіозв'язком. В даному випадку завмираннями можна рахувати зміну значень результатів сигналу при вбудовуванні інформації, атаках, додаванні шуму негауса і так далі Для боротьби із завмираннями в радіозв'язку використовують метод рознесення (по частоті, часу, простору, коду). Рознесення ЦВЗ по простору контейнера використовується у стеганографії.

IV клас: в кодері і в декодері додаткова інформація відома (обидва ключі замкнуті). Як відмічено в [9], перспективні стеганосистеми мають будуватися за даним принципом. Ця схема оптимальна шляхом узгодження кодера з сигналом-контейнером, та умовах спостереження каналу атак з адаптивним декодером управління.

3) В радіозв'язку важливим пристроєм є приймач, а стеганодетектор є головним в стеганосистемі. Він видає двійкові або М-ічні рішення про наявність/відсутності ЦВЗ (у разі детектора з м'якими рішеннями) залежно від типу. Розглянемо простий варіант "жорсткого" детектора таємної інформації. Операцію детектування позначимо через D. Тоді

$$D: I_w{}^* \times K^* \to \{0,1\}, \quad D(I_w, W) = D(I_w, F(I_w, K)) = \begin{cases} 1, & \text{ÿêùî} \quad W \quad {}^o \\ 0, & \text{ÿêùî} \quad W \quad \text{íåìà}^o \end{cases}. \qquad (1.6)$$

Детектор ЦВЗ використовують кореляційний приймач, показаний на рис.1.5.

Нехай у половини пікселів картинки результат яскравості збільшений на 1, а у інших - незмінним, чи зменшений на 1. Тоді $I_w=I_0+W$, де $F(I_0,K)=W$. Корелятор детектора ЦВЗ вираховує величину. $I_w \cdot W = (I_0 + W) \cdot W = I_0 \cdot W + W \cdot W$. Оскільки W здатне набувати значення ± 1, то $I_0 \cdot W$ буде не дуже багато, а $W \cdot W$ завжди позитивне. Тому $I_w \cdot W$ дуже близьке до $W \cdot W$. Тоді можна використати результати теорії зв'язку і записати ймовірність невірно виявленої закритої інформації, як додаткову функцію помилок від квадратного кореня з відношення $W \cdot W$ до дисперсії результатів пікселів яскравості ("енергія шуму").

При випадку м'якого детектора і невідкритої стегосистеми є дві основні міри подібності :

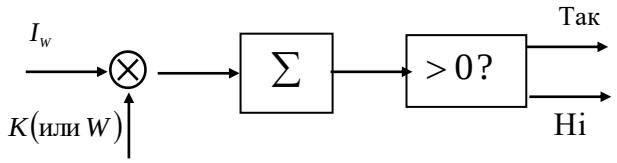


Рис.1.5. Кореляційний детектор ЦВЗ

$$\delta = \frac{I_0 I_W}{\|I_0\| \|I_W\|} \quad - \qquad (1.7)$$

нормований коефіцієнт взаємної кореляції і

$$\delta = N - \sum_{i=1}^{N} i_0 i_w - \qquad (1.8)$$

відстань по Хэммінгу.

В детекторі можливий варіант виникнення двох типів помилок. Є ймовірність, що детектор не виявить наявний ЦВЗ і ймовірність недійсного пошуку ЦВЗ в пустому контейнері (ймовірність неправдивої тривоги). Збільшення однієї ймовірності приводить до зменшення іншої. Ймовірність неправдивого виявлення характеризує надійність роботи детектора. Система ЦВЗ будується так, щоб зменшити ймовірність виникнення обох помилок, тому, ща кожна з них приводить до відмови в обслуговування.

3.2. Проектування структури бази даних

Список документів для функцій що автоматизуються.

Документи, котрі використовуються в функції «Додавання зображень до БД» показані в таблиці 3.1:

Таблиця 3.1

Інформаційний список документів

Код документа	Назва	Вхідн./вихідн.
DC-01	Розпорядження про додавання зображення до бази даних	вхідний

Документи, котрі використовуються в функції «Додавання повідомлень» подані в таблиці 3.2:

Таблиця 3.2

Інформаційний список документів

Код документу	Назва	Вхідний/вихідний
DC-02	Розпорядження про додавання повідомлення	вхідний

Розробка моделей даних для автоматизуємих функцій

Для виконання функції «Вбудовування інформації в зображення методом найменш значущого біту» використовуються наступні сутності, наведені на рис.3.1.

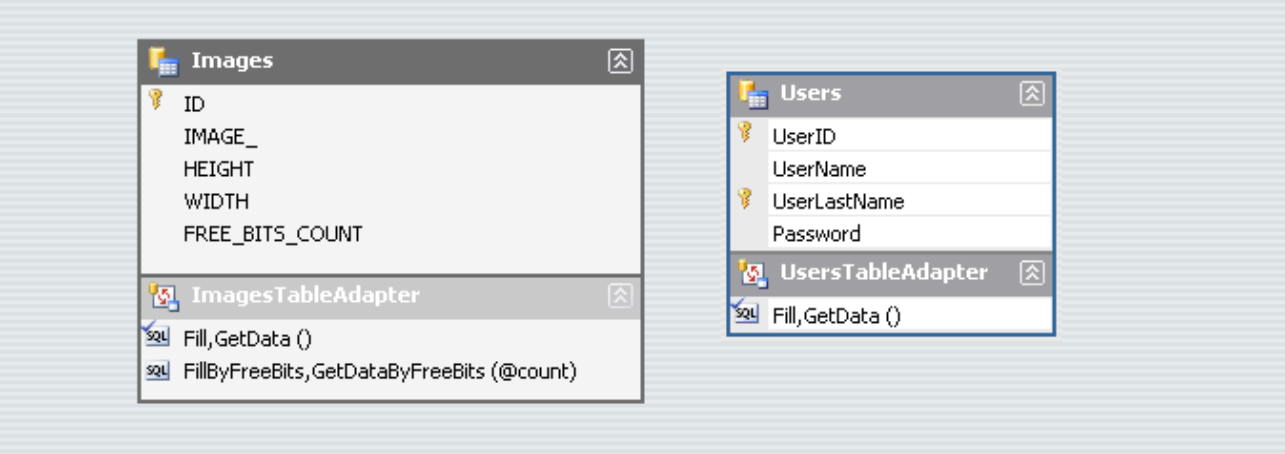


Рис.3.1. Сутність, використання в функції «Вбудовування інформації в зображення методом найменш значущого біту».

Обмеження атрибуту сутностей наведено в таблицях

Таблиця 3.3

Обмеження атрибутів сутності «Images»

№ п/п	Ім’я атрибуту	Тип	Розмір	Межі або допустимі значення	Структура (формат)	Умова	Величини за замовчуванням
1	ID	Рядок цифр	<=65535	0..9		Первинний ключ	null
2	IMAGE_	Зображення	**	**			null
3	HEIGHT	Рядок букв	<=65535	0..9			null
4	WIDTH	Рядок цифр	<=65535	0..9			null
5	FREE_BITS_COUNT	Рядок цифр	<=65535	0..9			null

Таблиця 3.4

Обмеження атрибутів сутності «Users»

№ п/п	Ім’я атрибуту	Тип	Розмір	Межі або допустимі значення	Структура (формат)	Умова	Величини за замовчуванням
1	UserID	Рядок цифр	<=65535	0..9		Первинний ключ	null
2	UserName	Рядок букв	<=50	А…Я,A..Z, 0..9			null
3	UserLastName	Рядок букв	<=50	А…Я,A..Z,0..9		Первинний ключ	null
4	Password	Рядок цифр	<=65535	0..9			null

В ході розробленнч БД створено наступні сутності: Images, users. Які потрібні для зручної роботи з додатком, який відповідає предметній області. Цілісність забезпечена за допомогою зовнішніх ключів також функцій БД оновлення та каскадного видалення.

Сутність «Images» використовується для додавання та редагування зображень.

Сутність «users» зберігає дані про користувачів системи та обмежує доступу до додатку програми.

На рис.3.2. та 3.3. зображена фізична і логічна модель бази даних.

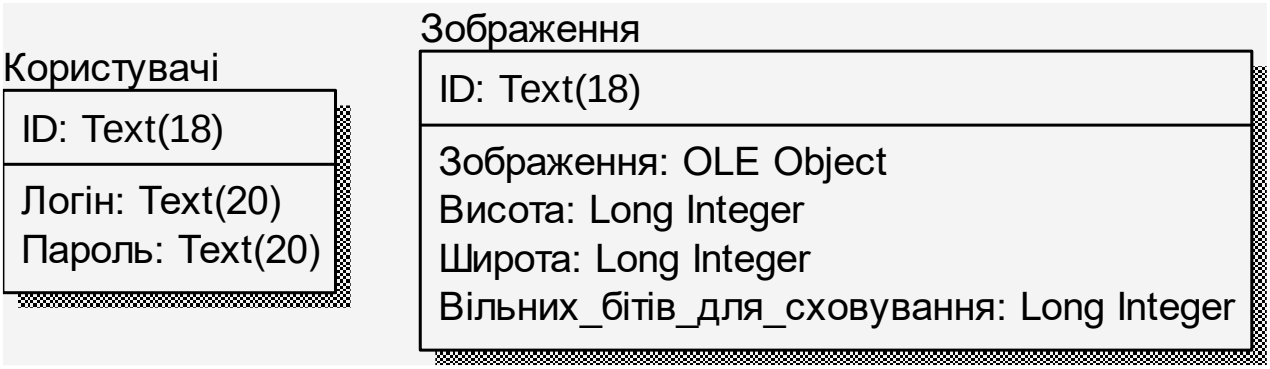


Рис.3.2. Фізична модель бази даних

Рис.3.3. Логічна модель бази даних

3.3. Опис архітектури додатку.

3.3.1 Розроблення діаграми класів, що реалізують бізнес-логіку програмної системи.

Клас «frmLoginForm» призначений для контролю доступу до даних користувачів. У даному класі викликаються методи перевірки наявності користувача з введеним логіном та паролем у базі даних.

Клас «MainForm» призначений для роботи з головною формою додатку. він являє собою вхідною точною у систему. З даного класу викликаються основні класи системи.

Клас «Steganography» призначений для виконання алгоритму приховування інформації методом найменш значущого біту.

Клас «HideParam» призначений для перевірки параметрів приховування та виконання приховування.

Клас «ExtractParam» призначений для отримання прихованого повідомлення.

Клас «StegoDBDataSet» призначається для роботи з базою даних. В ньому описано методи для роботи з інформацією у базі даних а саме додавання, редагування, видалення, пошуку та вибірки даних.

Лістинг програми наведено у додатку А.

Діаграма класів наведена в додатку Б рис Б.1.

3.3.2. Розроблення діаграми використання елементів графічного інтерфейсу користувача.

Робота з програмою починається з форми авторизації. Після вводу логіна та паролю здійснюється їх перевірка в базі даних. Якщо користувач ввів правильні дані, то він отримає доступ до головної форми додатку.

З головної форми користувач може створити нове повідомлення в ручну або відкрити з файлу, також є можливість зберегти його. Для цього можна використати нижнє меню або в головному меню натиснути «Файл». Обрати контейнер в ручну за допомогою кнопки «Открыть», або підібрати з бази данних натиснувши для цього кнопку «Подобрать из БД». Сховати інформацію і зберегти заповнений контейнер.

Для виклку довідки необхідно у головному меню обрати пункт «Помощь» та у ньому обрату пункт «Справка». Після цього з`явится керівництво користувача.

Для закінчення роботи з програмою необхідно вибрати пункт меню «Файл», «Выход».

Діаграма використання наведена у додатку В на рис. В.1.

3.4. Тестування додатку.

У даному розділі буде описано процедури тестування додатку. для перевірки надійності розроблюваної системи або відповідності його функціональним вимогам необхідно виконати тестування. Принциповим для даного програмного продуту є системне модульне та інтеграційне тестування.[13]

Тестові вимоги:

Для того, щоб упевнитися у тому, що програмний продукт «Модуль приховування інформації методом найменш значущого біту» задовольняє вимогам, вказаним у специфікації вимог «Модуль приховування інформації методом найменш значущого біту» необхідно протестувати наступні вимоги:

1. авторизація користувачів в системі;

2. додавання зображень до бази зображень;
3. видалення зображень з бази зображень;
4. приховування повідомлення в зображення;
5. отримання повідомлення із зображення;
6. відповідність ключів.

Група тестів: Робота з модулем

Тестовий приклад: № 1

Призначення: перевірка користувача зареєстрованого в системі.

Тест-вимоги, що перевіряються: 1

Тестові передумови: при авторизації користувачів система перевіряє логін та пароль.

Тест пройдено: реальні значення співпадають з очікуваними.

Група тестів: Робота з БД

Приклад: № 2

Призначення: перевірка того, що користувач має можливість додавати данні до БД.

Тест-вимоги, що перевіряються: 2

Тестові передумови: користувач повинен бути авторизований і вибрати необхідні зображення.

Критерій проходження: реальні значення збігаються з очікуваними.

Група тестів: Робота з БД

Приклад: № 3

Призначення: перевірка того, що користувач має можливість видалити данні з БД.

Тестові вимоги, що перевіряються: 3

Тестові передумови: користувач повинен бути авторизований і вибрати необхідні зображення.

Проходження тесту: реальні значення збігаються з очікуваними.

Група тестів: Робота з модулем.

Приклад: № 4

Призначення: перевірка того, що користувач має можливість приховувати обрану інформацію в обраник контейнер.

Тест-вимоги, що перевіряються: 4

Тестові передумови: користувач авторизований вводить повідомлення і вибирає контейнер (зображення).

Проходження тесту: реальні значення збігаються з очікуваними.

Група тестів: Робота з модулем.

Приклад: № 5

Призначення: перевірка того, що користувач має можливість отримувати приховану інформацію із заповненого контейнеру.

Тестові вимоги, що перевіряються: 5

Тестові передумови: користувач повинен бути авторизований і вибрати заповнений контейнер (зображення).

Проходження тесту: реальні значення збігаються з очікуваними.

Група тестів: Робота з модулем.

Тестовий приклад: № 6

Призначення: перевірка того, що користувач не маючі вірного ключа не зможе прочитати приховане повідомлення.

Тест-вимоги, що перевіряються: 6

Передумови для тесту: користувач повинен бути авторизований і вибрати заповнений контейнер (зображення) і ввести вірний і невірний ключ.

Критерій проходження тесту: всі реальні значення збігаються з очікуваними.

Тестових прикладів виконано: 6

Тестових прикладів пройдено: 6

Таблиці тестових сценаріїв наведені у додатку Г.

3.5. Розгортання програмного продукту.

3.5.1. Системні мінімальні характеристики

У цьому розділі наведено апаратні вимоги до забезпечення клієнтської частини. Щоб програма функціонувала необхідна наступна мінімальна конфігурація ПК:

а) 256 Мбайт ОЗУ;
б) 20 Мбайт вільного дискового простору;
в) відеокарта з підтримкою дозволу не менше 800 х 600 і можливістю відображення не менше 256 кольорів;
г) монітор SVGA;
д) процесор 800 МГц або вище.

3.5.2. Вимоги до програмного забезпечення клієнтської частини.

На ПК необхідно встановити наступне програмне забезпечення:

а) операційна система Windows 98, Windows ME, Windows 2000 (Service Pack 2 або вище), Windows XP, Vista, Windows 2003, Windows 7
б) платформа .Net версія 2.0 і вище

3.5.3. Спосіб виклику програми, запуск програми.

Так як було розроблено Windows додаток, для запуску необхідно зробити подвійний натискання клавіші миші по ярлику програми або безпосередньо по самому *. exe файлу.

РОЗДІЛ 4
ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Охорона праці

Тема кваліфікаційної роботи магістра присвячена розробці модуля забезпечення приховування даних на основі стеганографічного методу найменш значущого біту. Оскільки, розробка програми передбачає використання електронно-обчислювальної техніки, то важливим є дотримання вимог з охорони праці і техніки безпеки. Проаналізуємо основні правила і норми, яких необхідно дотримуватись при експлуатації комп'ютерів та периферійних пристроїв.

Охорона життя і здоров'я людини є пріоритетом соціальної політики держави. В Україні прийнято закон про пряму дію «Про охорону праці», який регулює захист конституційного права працівників на безпечні умови праці. Українське законодавство про охорону праці складається із загальних законів України та спеціальних законодавчих актів. Загальними законами України, що визначають основні положення з охорони праці, є Конституція України, Закон України «Про охорону праці», Кодекс законів про працю (КЗпП), Закон України «Про загальнообов'язкове державне соціальне страхування від Нещасні випадки 105 та професійні захворювання, що спричинили інвалідність.

Залежно від вимог до робочого місця, розмір одного робочого місця повинен бути не менше 6 кв. метрів. При необхідності сусідні комп'ютерні робочі місця слід розділити перегородками висотою до 2 метрів. При визначенні достатньої площі кімнати та робочого місця на одну особу необхідно враховувати шафи, сейфи, шафи чи інші предмети меблів чи обладнання в приміщенні.

На столі можна розміщувати допоміжні пристрої (принтери, колонки, сканери), а також зберігання документів, якщо це не обмежує видимість екрану та не заважає працівнику. У разі надмірного шуму або вібрації технічного обладнання роботодавець зобов'язаний забезпечити працівників антивібраційними килимками.

Робочий стілець повинен бути поворотним, легко регулюватися по висоті та забезпечувати достатню підтримку та зручне положення спини та спини.

Щодня проводити вологе прибирання, очищати робоче місце та монітор комп'ютера від пилу.

Загалом під терміном охорони праці в комп'ютерних системах розуміється відповідність усім вимогам і нормам, встановленим законодавством про охорону праці. Законодавство цього регіону зосереджено на якісній та безпечній експлуатації робочих приміщень та приміщень, дотриманні гігієнічних та гігієнічних умов праці та захисту від інших небезпечних факторів на підприємстві. Основне законодавство про охорону праці приділяє велику увагу покращенню умов праці в усіх галузях народного господарства, запровадженню сучасних заходів безпеки та забезпеченню гігієнічних та гігієнічних умов, що запобігають нещасним випадкам на виробництві та професійним захворюванням.

При підборі приміщень для робочих місць ПК враховується ступінь відбиття світла на екранах, що проходить через вікна, що може спричинити значну сліпоту у тих, хто сидить перед ними, особливо влітку та в сонячні дні. Тому комп'ютери та оргтехніка розташовують біля стін, яких немає біля вікон або перед ними.

Оскільки погане освітлення знижує продуктивність користувачів ПК та можливі негативні наслідки для здоров'я, такі як короткозорість та втома, усі кімнати, обладнані робочими станціями ПК, мають як природне, так і штучне освітлення. Розміщення робочих місць з ПК у підвалі не допускається.

Щоб екрани ПК не висвітлювалися прямими світловими потоками, лінії світильників розташовані з достатнім бічним зміщенням відносно рядів робочих місць, а також паралельно світловим отворам. При цьому на кожному вікні повинні бути світлорозсіюючі штори з коефіцієнтом відбиття 0,7.

Штучне внутрішнє освітлення слід передбачати у вигляді комбінованої системи освітлення з використанням люмінесцентних джерел світла в загальних світильниках, які розміщують над робочими поверхнями в рівному і прямокутному порядку. Штучне освітлення забезпечує освітлення робочих місць ПК 300 - 500 люкс.

Отже, при розробці модуля забезпечення приховування даних на основі стеганографічного методу найменш значущого біту, проаналізовано та враховано необхідні вимоги щодо охорони праці при використанні електронно-обчислювальної техніки і забезпечено умови для зручної та ефективної роботи працівників.

4.2 Підвищення стійкості роботи об'єктів господарської діяльності в воєнний час

Основні шляхи підвищення стійкості роботи промислових об’єктів у надзвичайних умовах мирного і воєнного часів:

- забезпечення надійного захисту робітників і службовців від ЗМУ (засобів масового ураження);
- захист виробничих фондів від вражаючих факторів ЗМУ, в тому числі і від вторинних;
- підвищення надійності і оперативності управлінням виробництвом і ЦЗ;
- забезпечення стійкості постачання підприємства електроенергією, газом, водою та інше;
- підготовка об’єкта до проведення відновлювальних робіт.

Підвищення стійкості роботи промислових підприємств в умовах НС мирного і воєнного часів досягається завчасним проведенням комплексу інженерно-технічних, технологічних і організаційних заходів.

Інженерно-технічні заходи (ІТЗ) включають комплекс робіт по підвищенню міцності і надійності будинків, споруд комунально-енергетичних систем, матеріально-технічних запасів.

Технологічні заходи спрямовані на підвищення стійкості виробництва шляхом заміни існуючого технологічного режиму роботи на такий, що виключає можливість виникнення вторинних вражаючих факторів.

Організаційні заходи передбачають розробку і планування дій в умовах НС керівного складу об’єкту, штабу, служб та невоєнізованих формувань ЦЗ по захисту робітників і службовців, проведення рятувальних робіт та відновлення порушеного виробництва.

Підвищення стійкості об'єкта досягається посиленням найбільш слабких (вражаючих) елементів і ділянок об'єкта. Для цього на кожному ОГД завчасно на основі досліджень планують і проводять відповідні організаційні й інженерно-технічні заходи.

Досягнення науки і техніки дозволяють реалізувати такі рішення, при яких підприємство буде стійке до впливу дуже значних надлишкових тисків, однак це пов'язано з великими витратами засобів і матеріалів і може бути виправдано лише при захисті унікальних, особливо важливих елементів об'єкта. Заходи будуть економічно обґрунтовані, якщо вони максимально узгоджені із завданнями, які розв'язуються в мирний час для забезпечення безаварійної роботи, поліпшення умов праці, удосконалювання виробничого процесу. Тому підвищення характеристик міцності проводять, якщо:

- окремі особливо важливі будинки і спорудження значно слабші за інші і їхню міцність доцільно довести до прийнятої для даного підприємства межі стійкості;

- необхідно зберегти деякі важливі ділянки (цехи), які можуть самостійно функціонувати при виході з ладу інших і забезпечать випуск особливо цінної продукції.

Особливо велике значення має розробка інженерно-технічних заходів при новому будівництві, бо у процесі проектування, як відзначалося раніше, у багатьох випадках можна домогтися логічного поєднання загальних інженерних рішень із захисними заходами ЦЗ, що знизить витрати на їх реалізацію.

На існуючих об'єктах заходи щодо підвищення стійкості доцільно проводити в процесі реконструкції чи виконання інших ремонтно-будівельних робіт.

Підвищення стійкості роботи промислових об'єктів передбачає:

- захист робітників та службовців у надзвичайних ситуаціях мирного і воєнного часу;

- підвищення міцності і стійкості найважливіших елементів і удосконалювання технологічного процесу;

- підвищення стійкості матеріально-технічного постачання;

- підвищення стійкості управління об'єктом;

- розробку заходів щодо зменшення імовірності виникнення вторинних факторів ураження і збитків від них;

- підготовку до відновлення виробництва після ураження об'єкта.

Особлива увага повинна бути приділена забезпеченню укриттям всіх працюючих у захисних спорудженнях. З цією метою розробляється план нагромадження і будівництва необхідної кількості захисних споруджень; у випадку недостачі сховищ, які відповідають сучасним вимогам, у ньому передбачається укриття робітників та службовців у швидкостворюваних сховищах.

При проектуванні і будівництві нових цехів підвищення стійкості може бути досягнуто застосуванням для несучих конструкцій високоміцних і легких матеріалів (легованих сталей, алюмінієвих сплавів).

При будівництві і реконструкції промислових споруд необхідно застосовувати легкі, вогнестійкі покрівельні матеріали, полегшені міжповерхові перекриття і сходові марші, підсилюючи їх кріплення до балок. Обвалення цих

матеріалів і конструкцій принесе меншу шкоду устаткуванню, ніж важких залізобетонних.

Заходи щодо підвищення стійкості технологічного і верстатного устаткування повинні бути спрямовані на забезпечення його збереження для випуску продукції після надзвичайної ситуації. Однак підвищити стійкість устаткування можна, підсилюючи його найбільш слабкі елементи і створюючи запаси цих елементів, окремих вузлів і деталей, матеріалів та інструментів для ремонту і відновлення пошкоджень.

Важке устаткування розміщують, по можливості, на нижніх поверхах виробничих будівель. Велике значення має міцне закріплення на фундаментах верстатів і установок, які мають велику висоту і малу площу опори; використання розтяжок і додаткових опор підвищить їх стійкість до перекидання. Прилади бажано встановлювати на закріплених підставках, тумбах, столах. Особливо цінне й унікальне устаткування потрібно розміщувати в заглиблених підземних чи спеціально побудованих приміщеннях підвищеної міцності і на випадок виникнення надзвичайних ситуацій розробити спеціальні індивідуальні енергогасильні пристрої.

При удосконалюванні технологічних процесів виробництва слід вживати і заходи для підвищення їх стійкості, пам'ятаючи, що найбільш важливі умови надійності - стійкість системи управління і безперебійність забезпечення усіма видами енергопостачання. У випадку виходу з ладу автоматичних систем управління повинен бути передбачений перехід на ручне управління процесом у цілому чи окремими його ділянками.

Підвищення стійкості технологічного процесу досягається розробкою способів продовження виробництва при виході з ладу окремих верстатів, ліній і навіть окремих цехів за рахунок переведення виробництва в інші цехи; розміщенням виробництва окремих видів продукції у філіях; шляхом заміни зразків, устаткування, що вийшли з ладу, іншими; а також скороченням числа використовуваних типів верстатів і приладів.

На випадок значних руйнувань повинна бути передбачена заміна складних технологічних процесів більш простими з використанням найбільш стійких типів устаткування і контрольно-вимірювальних приладів, які збереглись. Необхідно заздалегідь розробити можливі зміни в технології з метою заміни дефіцитних матеріалів, деталей і сировини на більш доступні.

На всіх об'єктах розробляються способи безаварійної зупинки виробництва за сигналом "Повітряна тривога" ("ПТ"). У кожній зміні призначаються люди, які повинні відключати джерела живлення і технологічні

установки. Якщо за умовами технологічного процесу зупинити окремі ділянки виробництва, агрегати, печі і т.п. не можна, їх переводять на знижений режим роботи; ті, що спостерігають за безупинною роботою цих елементів, повинні бути забезпечені індивідуальними укриттями, спорудженими в безпосередній близькості від робочого місця.

Підвищення стійкості системи енергопостачання досягається проведенням як загальноміських, так і об'єктових інженерно-технічних заходів. Створюються дублюючі джерела електроенергії, газу, води і пари шляхом прокладання декількох електро-, газо-, водо- і паропостачальних комунікацій та подальшого їх закільцювання. Інженерні й енергетичні комунікації переносяться в підземні колектори, найбільш відповідальні пристрої (центральні диспетчерські розподільні пункти) розміщуються в підвальних приміщеннях будинків чи у спеціально побудованих міцних спорудах. Там, де прокладання комунікацій у траншеях чи тунелях неможливе, здійснюється закріплення трубопроводів до естакад, щоб уникнути їх зрушення чи скидання; самі естакади зміцнюються установкою розтяжок у місцях поворотів і розгалужень.

Стійкість систем електропостачання об'єкта підвищують, підключаючи його до декількох джерел живлення, віддалених одне від одного на відстань, що виключає можливість їх одночасного ураження одним ядерним вибухом.

Для стійкого і надійного постачання підприємств газом необхідно передбачити його подачу в газові мережі об'єктів від газорегуляторних пунктів (газороздавальних станцій), а на випадок виходу з ладу останніх влаштовувати обвідні лінії - байпаси. При будівництві нових чи реконструкції старих газових мереж по можливості повинні створюватися закільцьовані системи. Усі вузли і лінії газопостачання бажано розміщувати під землею (заглиблення комунікацій значно зменшує імовірність їх ураження ударною хвилею ядерного вибуху й інших засобів нападу, а крім того, значно знижує можливість виникнення вторинних факторів ураження).

З метою зменшення пожежної небезпеки (зниження можливості витікання газу) на газопроводах встановлюються автоматичні запірні і перемикаючі пристрої дистанційного керування, що дозволяють при розриві труб безпосередньо з диспетчерського пункту відключати мережі чи переключати потік газу.

Підвищення стійкості систем теплопостачання досягається захистом джерел тепла і заглибленням комунікацій у ґрунт. Якщо на об'єкті передбачається будівництво котельні, її доцільно розміщувати в спеціальній

будівлі, яка стоїть окремо. Будинок котельні повинен мати полегшене перекриття і легке стінове заповнення.

Заходи по підвищенню стійкості системи каналізації розробляють окремо для зливових, промислових і господарських (фекальних) зливів. На об'єкті обладнують не менше двох виводів з підключенням до міських каналізаційних колекторів і додатково обладнують виводи для аварійних скидань неочищених вод у прилеглі до об'єкта яри та інші природні заглиблення.

Одним із найважливіших заходів по забезпеченню сталого, безперервного на всіх етапах управління у надзвичайних ситуаціях є розподіл всього персоналу об'єкта на дві групи: працююча зміна (перебуває на об'єкті) і відпочиваюча (перебуває у заміській зоні або по дорозі між заміською зоною та об'єктом). До того ж створюються дві-три групи управління (за кількістю змін), які, крім керівництва виробництвом, повинні бути готові будь-якої миті взяти на себе організацію і керівництво проведенням рятувальних та ремонтних робіт.

У районі розосередження робітників і службовців також обладнують пункт управління. Між міським і заміським пунктами управління проводять зв'язок, як правило, телефонний, передбачаючи його дублювання за допомогою радіо- та пересувних засобів, також вживають заходів по забезпеченню зв'язку із змінними підприємствами по кооперації.

Особливе значення має сталість виробничих та господарських зв'язків з постачання об'єкта всіма видами енергії, водою, парою, газом; з транспортних послуг; з поставок сировини, напівфабрикатів, комплектуючих виробів та ін.

Підвищення сталості матеріально-технічного постачання забезпечується створенням запасів сировини, матеріалів, комплектуючих виробів, обладнання, палива. Розміри незменшуваних запасів визначають для кожного об'єкта залежно від можливості їх накопичення, важливості продукції, яка випускається, визначених термінів переходу на виробництво продукції в умовах надзвичайних ситуацій.

Стабільно працююче підприємство повинно бути здатним безперебійно випускати продукцію за рахунок наявних запасів до відновлення зв'язків з поставок або до одержання необхідного від нових постачальників.

ВИСНОВКИ

В даній магістерській роботі було описано напрями діяльності та організаційна структура підприємства «Trade-box». Побудована схема бізнес-процесів підприємства. Проведене порівняння функціональності існуючих програмних продуктів, що реалізують приховування інформації методом найменш значущого біту. Розроблено вимоги до створюваного програмного продуту: глосарій, діаграма варіантів використання, специфікація варіантів використання, розкадровка, специфікація функціональних та не функціональних вимог. Також описано математичну постановку задачі, спроектовано базу даних, проведено тестування додатку. Проведена робота повністю відповідає завданню до магістерської роботи.

Створено програмний продукт який реалізує приховування інформації в зображенні методом найменш значущого біту. Створений продукт не поступається своїми якісними показниками відомим програмним продуктам, таким як ImageSpyer і Steganos Security Suite.

При виконанні магістерської роботи було отримано наукові та практичні навики створення програмних продуктів, оформлення звітності, розробка вимог до програмних продуктів.

Розроблений програмний продукт може використовуватись для прихованої передачі даних без видимого факту передачі. Також для підтвердження власника зображення, за допомогою розробленого програмного продукту можна в зображення додати цифровий водяний знак.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Быков С.Ф. Алгоритм сжатия JPEG с позиций компьютерной стеганографии. Защита информации: Конфидент. 2000. –432с.
2. Чиссар И., Кернер Я. Теория информации: Теоремы кодирования для дискретных систем без памяти / Перевод с англ. - М.: Мир, 1985, –400 с.
3. Яковлев В.А. Защита информации на основе кодового зашумления. Часть 1. Теория кодового зашумления. / Под ред. В.И. Коржика.– С.Пб.: ВАС, 1993.–245с.
4. Грибунин В.Г., Оков И.Н., Туринцев И.В. Стеганографическая защита речевых сигналов в каналах открытой телефонной связи / Сборник тезисов Российской НТК “Методы и технические средства обеспечения безопасности информации”, – СПб.:, ГТУ, 2001, 83-84с.
5. Теория электрической связи: Учебник для вузов / Зюко А.Г., Кловский Д.Д., Коржик В.И., Назаров М.В. – М.: Радио и связь, 1999.– 432с.
6. Алферов А.П., Зубов А.Ю., Кузьмин А.С., Черемушкин А.В. Основы криптографии.–М.: Гелиус АРВ, 2001.– 480 с.
7. Arnold M., Kanka S. MP3 robust audio watermarking // International Watermarking Workshop. 1999. – 548с.
8. Оков И.Н., Ковалев Р.М. Электронные водяные знаки как средство аутентификации передаваемых сообщений // Защита информации. Конфидент. 2001. № 3, 80-85с.
9. Оков И.Н. Криптографические системы защиты информации. – СПб.:, ВУС, 2001. –236с.
10. Оков И.Н., Ковалев Р.М. Электронные водяные знаки как средство аутентификации передаваемых сообщений // Защита информации. Конфидент. 2001. № 3, 80-85с.
11. Шеннон К. Работы по теории информации и кибернетики / Пер. с англ. – М.: Иностранная литература, 1963. – 829с.

ДОДАТКИ

ДОДАТОК А

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ІВАНА ПУЛЮЯ

М А Т Е Р І А Л И

ІХ НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

8–9 грудня 2021 року

ТЕРНОПІЛЬ
2021

УДК 004.056
О.Р. Кучма
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

РОЗРОБЛЕННЯ МОДУЛЯ ЗАБЕЗПЕЧЕННЯ ПРИХОВУВАННЯ ДАНИХ НА ОСНОВІ СТЕГАНОГРАФІЧНОГО МЕТОДУ НАЙМЕНШ ЗНАЧУЩОГО БІТУ

UDC 004.056
O.R. Kuchma

DEVELOPMENT OF A DATA HIDING MODULE BASED ON STEGANOGRAPHIC METHOD OF THE LEAST SIGNIFICANT BIT

Комп’ютерні технології надали нового імпульсу розвитку і вдосконаленню стеганографії, з’явився новий напрям в галузі захисту інформації – цифрова стеганографія (ЦС). На відміну від криптографії, методи стеганографії приховують сам факт передачі інформації. Безпека методів грунтується на збереженні стеганографічних перетворенням основних властивостей відкритого файлу, що передається при внесенні до нього таємного повідомлення і деякої невідомої супротивникові інформації – ключа. Основним завданням стеганографії є подолання систем моніторингу та управління мережевими ресурсами. Стеганографічні методи, спрямовані на протидію систем моніторингу та управління мережевими ресурсами промислового шпигунства, дозволяють протистояти спробам контролю над інформаційним простором при проходженні інформації через сервери управління локальних і глобальних обчислювальних мереж. Іншим важливим завданням стеганографії є камуфлюванні програмного забезпечення (ПЗ). У тих випадках, коли використання ПЗ не зареєстрованими користувачами є небажаним, воно може бути закамуфльоване під стандартні універсальні програмні продукти (наприклад, текстові редактори) або приховано у файлах мультимедіа (наприклад, у звуковому супроводі комп’ютерних ігор). Ще однією областю використання стеганографії є захист авторського права від піратства. На комп’ютерні графічні зображення наноситься спеціальна позначка, яка залишається невидимою для очей, але розпізнається спеціальним ПЗ. Таке програмне забезпечення вже використовується в комп’ютерних версіях деяких журналів. Даний напрямок стеганографії призначене не тільки для обробки зображень, але й для файлів з аудіо і відео наповненням покликане забезпечити захист інтелектуальної власності.

Література.

1. Стеганографічні методи захисту документів / Б. В. Дурняк, Д. В. Музика, В. І. Сабат. – Львів : Укр. акад. друкарства, 2014. – 159 с. : іл., портр. ; 21 см. – На паліт.: Інформ. технології. – Частина тексту парал. укр., англ. – Бібліогр.: с. 149–159 (118 назв). – 300 пр. – ISBN 978-966-322-401-5
2. Конахович Г. Ф., Пузиренко О. Ю. Комп’ютерна стеганографія. Теорія і практика. – К.: «МК-Пресс», 2006. – 288 с., іл.

ДОДАТОК Б

Лістинг програми.

MainForm.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Diagnostics;
using System.IO;
using System.Drawing.Imaging;

namespace PROEKT
{
   public partial class MainForm : Form
   {
      /// <summary>
      ///
      /// </summary>
      /// <param name="byteArrayIn"></param>
      /// <returns></returns>
      public Bitmap ByteArrayToImage(byte[] byteArrayIn)
      {
         MemoryStream ms = new MemoryStream(byteArrayIn);
         Bitmap returnImage = (Bitmap)Bitmap.FromStream(ms);
         return returnImage;
      }

      public static string keyStart, keyEnd;
      public MainForm()
      {
         InitializeComponent();
      }
      //
      private void открытьИзображениеToolStripMenuItem_Click(object sender, EventArgs e)
      {
         if (OpenPictureDialog.ShowDialog() == DialogResult.OK)
         {
            ContainerBox.Load(OpenPictureDialog.FileName);
            SizeStatusLabel.Text ="Размер " +  ContainerBox.Image.Width.ToString() + " x " + ContainerBox.Image.Height.ToString() + " точек";
            HideKolStatusLabel.Text =" Доступно для скрытия " + Convert.ToString(ContainerBox.Image.Width * ContainerBox.Image.Height * 3)
+ " бит";
         }
      }
      //
      private void скрытьToolStripMenuItem_Click(object sender, EventArgs e)
      {
         if (ContainerBox.Image != null)
         {
            if (MesageBox.Text != "")
            {
               HideParam paramForm = new HideParam(ContainerBox.Image, MesageBox.Text);

               if (paramForm.ShowDialog() == DialogResult.OK)
               {
                  ContainerBox.Image = Steganography.GetDataBitmap();
                  MesageBox.Text = "";
               }
            }
            else
               MessageBox.Show("Введите хоть какое нибудь сообщение!", "Warning", MessageBoxButtons.OK, MessageBoxIcon.Warning);
         }
         else
            MessageBox.Show("Загрузите изображение!", "Warning", MessageBoxButtons.OK, MessageBoxIcon.Warning);

      }
      //
      private void извлечСообщениеToolStripMenuItem_Click(object sender, EventArgs e)
      {
         if (ContainerBox.Image != null)
         {
```

```
            ExtractParam extractForm = new ExtractParam(ContainerBox.Image);
            if (extractForm.ShowDialog() == DialogResult.OK)
            {
                MesageBox.Text = extractForm.GetMessage();
            }
        }
        else
            MessageBox.Show("Загрузите изображение!", "Warning", MessageBoxButtons.OK, MessageBoxIcon.Warning);

    }
    //
    private void создатьНовоеСообщениеToolStripMenuItem_Click(object sender, EventArgs e)
    {
        MesageBox.Text = "";
    }
    //
    private void загрузитьСообщеToolStripMenuItem_Click(object sender, EventArgs e)
    {
        if (OpenTextDialog.ShowDialog() == DialogResult.OK)
        {
            StreamReader sr = new StreamReader(OpenTextDialog.FileName, Encoding.GetEncoding(1251));
            MesageBox.Text = sr.ReadToEnd();
            sr.Close();
        }
    }
    // Процедура сохранения сообщения в файл
    private void сохранитьСообщениВФайлToolStripMenuItem_Click(object sender, EventArgs e)
    {
        if (SaveTextDialog.ShowDialog() == DialogResult.OK)
        {
            StreamWriter sw = new StreamWriter(SaveTextDialog.FileName, false, Encoding.GetEncoding(1251));
            sw.Write(MesageBox.Text);
            sw.Close();
        }
    }
    //
    private void сохранитьИзображениеToolStripMenuItem_Click(object sender, EventArgs e)
    {
        try
        {
            if (SavePictureDialog.ShowDialog() == DialogResult.OK)
            {
                Bitmap bm = new Bitmap(ContainerBox.Image);
                bm.Save(SavePictureDialog.FileName, ImageFormat.Bmp);
            }
        }
        catch (NullReferenceException)
        {

            MessageBox.Show("Пустоту сохранять нельзя!", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);;
        }
    }
    private void просмотрВОтдельномОкнеToolStripMenuItem_Click(object sender, EventArgs e)
    {
        PictureWindow pictForm = new PictureWindow(ContainerBox.Image);
        if (pictForm.ShowDialog() == DialogResult.OK)
        {
            ContainerBox.Image = pictForm.img;
            SizeStatusLabel.Text = "Размер " + ContainerBox.Image.Width.ToString() + " x " + ContainerBox.Image.Height.ToString() + " точек";
            HideKolStatusLabel.Text = " Доступно для скрытия " + Convert.ToString(ContainerBox.Image.Width * ContainerBox.Image.Height * 3)
+ " бит";
        }

    }

    private void редактироватьВОтToolStripMenuItem_Click(object sender, EventArgs e)
    {
        MessageWindow messForm = new MessageWindow(MesageBox.Text);
        if (messForm.ShowDialog() == DialogResult.OK)
        {
            MesageBox.Text = messForm.MSG;
        }
    }

    private void вЫХОДToolStripMenuItem_Click(object sender, EventArgs e)
    {
        Application.Exit();
    }
```

```
        private void копироватьToolStripMenuItem_Click(object sender, EventArgs e)
        {
            MesageBox.Copy();
        }

        private void вырезатьToolStripMenuItem_Click(object sender, EventArgs e)
        {
            MesageBox.Cut();
        }

        private void вставитьToolStripMenuItem_Click(object sender, EventArgs e)
        {
            MesageBox.Paste();
        }

        private void MesageBox_TextChanged(object sender, EventArgs e)
        {
            CurentKolStatusLabel.Text ="Текущее количество бит для скрытия(без учёта ключей) "+ Convert.ToString(MesageBox.Text.Length *
12);
        }

        private void MainForm_FormClosing(object sender, FormClosingEventArgs e)
        {
            Application.Exit();
        }



        private void помощьToolStripMenuItem_Click(object sender, EventArgs e)
        {
            try
            {
                Process.Start("help.chm");
            }
            catch (Exception)
            {
                MessageBox.Show("Что-то не впорядке с файлом справки", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }

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

        public struct BitmapInfo
        {
            public Bitmap bitmap;
            public string name;
        }

        private void mniImport_Click(object sender, EventArgs e)
        {
            OpenPictureDialog.Multiselect = true;
            if (OpenPictureDialog.ShowDialog() == System.Windows.Forms.DialogResult.OK)
            {
                List<BitmapInfo> list = new List<BitmapInfo>();
                for (int i = 0; i < OpenPictureDialog.FileNames.Length; i++)
                {
                    BitmapInfo inf = new BitmapInfo();
                    inf.bitmap = new Bitmap(OpenPictureDialog.FileNames[i]);
                    inf.name = OpenPictureDialog.FileNames[i];

                    list.Add(inf);
                }

                PrevForm frm = new PrevForm(list);
                frm.ShowDialog();
            }
        }

        private void toolStripButton11_Click(object sender, EventArgs e)
        {
            if (MesageBox.Text.Length > 0)
            {
                imagesTableAdapter1.FillByFreeBits(stegoDBDataSet1.Images, MesageBox.Text.Length * 12);
```

```
            if (stegoDBDataSet1.Images.Rows.Count > 0)
            {
                List<BitmapInfo> list = new List<BitmapInfo>();

                for (int i = 0; i < stegoDBDataSet1.Images.Rows.Count; i++)
                {
                    BitmapInfo inf = new BitmapInfo();
                    inf.bitmap = ByteArrayToImage(stegoDBDataSet1.Images[i].IMAGE_);
                    inf.name = stegoDBDataSet1.Images[i].ID.ToString();

                    list.Add(inf);
                }

                AvalPictForm frm = new AvalPictForm(list);
                if (frm.ShowDialog() == System.Windows.Forms.DialogResult.OK)
                {
                    this.ContainerBox.Image = frm.SelectedBitmap;
                }
            }
            else
            {
                MessageBox.Show("В базе данных нет подходящего изображения!", "Warning", MessageBoxButtons.OK,
MessageBoxIcon.Information);
            }
        }
        else
        {
            MessageBox.Show("Введите хоть какое нибудь сообщение!", "Warning", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        }
    }
  }
}
```

Steganogpaphy.cs

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Drawing;

namespace PROEKT
{
   class Steganography
   {
      /// <summary>
      /// Переменная содержащая изображение с данными
      /// </summary>
      private static Bitmap DataBitmap;
      /// <summary>
      /// Результат скрытия
      /// </summary>
      public enum HideRezult
      {
         hrToSmallPic, // Результат оповещающий о том что контейнера не достаточно для того что бы скрыть в нём информацию
         hrWrongPos, // Результат оповещающий о том что выбрана неверная позиция скрытия
         hrOk // Результат оповещающий об удачном скрытии
      }

      /// <summary>
      /// Возвращает изображение с данными
      /// </summary>
      /// <returns>Изображение</returns>
      public static Bitmap GetDataBitmap()
      {
         return DataBitmap;
      }

      /// <summary>
      /// Перевод числа из десятичной системы счисления в двоичную
      /// </summary>
      /// <param name="delimoe">Число которое нужно перевести</param>
      /// <param name="kol_razryadov">Нужное количество разрядов. Если не требуется то передавать 0</param>
      /// <returns>Двоичное число</returns>
      public static string DecToBin(int delimoe, int kol_razryadov)
      {
         int nach_delimoe = 0;
         string bin_v = "";
         int ostatok;
         // Цикл половинного деления
         while (delimoe > 1)
```

```
        {
            nach_delimoe = delimoe;
            delimoe = (int)delimoe / 2;
            ostatok = nach_delimoe - (delimoe * 2);
            bin_v += ostatok.ToString();
        }
        bin_v += delimoe.ToString();// Плюсуем последний бит
        // разворачиваем строку
        string temp_string = bin_v;
        bin_v = "";
        for (int i = temp_string.Length - 1; i >= 0; i--)
            bin_v += temp_string[i];
        // дописываем нужное количество разрядов
        if (kol_razryadov > bin_v.Length)
        {
            int dl = bin_v.Length;
            for (int i = 0; i < (kol_razryadov - dl); i++)
                bin_v = "0" + bin_v;
        }

        return bin_v;
    }

    /// <summary>
    /// Перевод числа из двоичной системы счисления в десятичную
    /// </summary>
    /// <param name="bin_value">Двоичное число</param>
    /// <returns>Десятичное число</returns>
    public static int BinToDec(string bin_value)
    {
        int dec = 0;
        int j = 0;
        for (int i = bin_value.Length - 1; i >= 0; i--)
        {
            dec += Int32.Parse((bin_value[i].ToString())) * Convert.ToInt32(Math.Pow(2, j));
            j++;
        }
        return dec;
    }

    /// <summary>
    /// Функция скрывает текстовое сообщение в изображение
    /// </summary>
    /// <param name="ContainerBMP">Изображение в которое нужно спрятать текстовое сообщение</param>
    /// <param name="messageString">Текстовое сообщение которое нужно скрыть</param>
    /// <param name="keyStart">Ключ начала полезного сообщения</param>
    /// <param name="keyEnd">Ключ конца полезного сообщения</param>
    /// <returns>Результат скрытия</returns>
    public static HideRezult HideData(Bitmap ContainerBMP, string messageString, string keyStart, string keyEnd)
    {
        int i, j;
        //Общее сообщение вместе с ключами
        string MESSAGE = keyStart + messageString + keyEnd;
        //Количество бит которые можно заменить
        int total = Convert.ToInt32(ContainerBMP.Width * ContainerBMP.Height);

        if ((MESSAGE.Length * 12) > (total * 3)) //Проверка, хватит ли нам картинки
        {
            DataBitmap = null; // Если не хватает картинки то возвращаем соотв. уведомление
            return HideRezult.hrToSmallPic;
        }
        else  // иначе если картинки хватает то
        {
            // Создаём изображение в котором будут хранится данные
            DataBitmap = new Bitmap(ContainerBMP.Width, ContainerBMP.Height);

            // Переменная которая будет давать нам знать заканчивать скрытие или нет
            bool flag = false;


            int n = 0, m = 0;// Позыции текущего бита сообщения

            string bitch;// Переменная которая содержит символ в двоичном виде
            Color col; // Структура цвета которую
            Color newColor;// Изменённая цветовая структура
            // Скрываем
            bitch = DecToBin(((int)MESSAGE[0]), 12);// Преобразуем первый символ

            j = 0;// Устанавливаем позицию Х-а
```

```
            int R, G, B;// Переменные содержащие цвета
            for (i = 0; i < ContainerBMP.Height; i++)
            {
                while (j < ContainerBMP.Width)
                {
                    col = ContainerBMP.GetPixel(j, i); // Считываем цветовую структуру по заданным координатам
                    R = col.R;
                    G = col.G;
                    B = col.B;
                    // Преобразуем КРАСНЫЙ цвет
                    if (bitch[m] == '0' && (R % 2 != 0))
                        R -= 1;
                    if (bitch[m] == '1' && (R % 2 == 0))
                        R += 1;
                    m++;// перебираем следующий битик
                    // Преобразуем ЗЕЛЁНЫЙ цвет
                    if (bitch[m] == '0' && (G % 2 != 0))
                        G -= 1;
                    if (bitch[m] == '1' && (G % 2 == 0))
                        G += 1;
                    m++;// перебираем следующий битик
                    // Преобразуем СИНИЙ цвет
                    if (bitch[m] == '0' && (B % 2 != 0))
                        B -= 1;
                    if (bitch[m] == '1' && (B % 2 == 0))
                        B += 1;
                    m++;// перебираем следующий битик
                    // копируем преобразованный цвет
                    newColor = Color.FromArgb(R, G, B);
                    ContainerBMP.SetPixel(j, i, newColor);

                    // смещаемся по сообщению
                    if (m == 12)
                    {
                        m = 0;
                        n += 1;
                        bitch = "";
                        if (n == MESSAGE.Length)
                        {
                            flag = true;
                            break;
                        }
                        else
                            bitch = DecToBin(((int)MESSAGE[n]), 12);
                    }
                    j++;


                }
                j = 0;
                if (flag)
                    break;
            }
            DataBitmap = ContainerBMP;
            return HideRezult.hrOk;

        }
    }
    /// <summary>
    ///
    /// </summary>
    /// <param name="ContainerBMP"></param>
    /// <param name="messageString"></param>
    /// <param name="keyStart"></param>
    /// <param name="keyEnd"></param>
    /// <param name="pos_X"></param>
    /// <param name="pos_Y"></param>
    /// <returns></returns>
    public static HideRezult HideData(Bitmap ContainerBMP, string messageString, string keyStart, string keyEnd, int pos_X, int pos_Y)
    {
        int i, j;
        //Общее сообщение вместе с ключами
        string MESSAGE = keyStart + messageString + keyEnd;
        //Количество бит которые можно заменить
        int total = Convert.ToInt32(ContainerBMP.Width * ContainerBMP.Height - (pos_Y * ContainerBMP.Width + pos_X));

        if ((MESSAGE.Length * 12) > (total * 3)) //Проверка, хватит ли нам картинки
        {
```

```
    DataBitmap = null; // Если не хватает картинки то возвращаем соотв. уведомление
    return HideRezult.hrToSmallPic;
}
else  // иначе если картинки хватает то
{
    // Создаём изображение в котором будут хранится данные
    DataBitmap = new Bitmap(ContainerBMP.Width, ContainerBMP.Height);

    // Переменная которая будет давать нам знать заканчивать скрытие или нет
    bool flag = false;


    int n = 0, m = 0;// Позыции текущего бита сообщения

    string bitch;// Переменная которая содержит символ в двоичном виде
    Color col; // Структура цвета которую считываем
    Color newColor;// Изменённая цветовая структура
    // Скрываем
    bitch = DecToBin(((int)MESSAGE[0]), 12);// Преобразуем первый символ

    j = pos_X;// Устанавливаем позицию X-а

    int R, G, B;// Переменные содержащие цвета
    for (i = pos_Y; i < ContainerBMP.Height; i++)
    {
        while (j < ContainerBMP.Width)
        {
            col = ContainerBMP.GetPixel(j, i); // Считываем цветовую структуру по заданным координатам
            R = col.R;
            G = col.G;
            B = col.B;
            // Преобразуем КРАСНЫЙ цвет
            if (bitch[m] == '0' && (R % 2 != 0))
                R -= 1;
            if (bitch[m] == '1' && (R % 2 == 0))
                R += 1;
            m++;// перебираем следующий битик
            // Преобразуем ЗЕЛЁНЫЙ цвет
            if (bitch[m] == '0' && (G % 2 != 0))
                G -= 1;
            if (bitch[m] == '1' && (G % 2 == 0))
                G += 1;
            m++;// перебираем следующий битик
            // Преобразуем СИНИЙ цвет
            if (bitch[m] == '0' && (B % 2 != 0))
                B -= 1;
            if (bitch[m] == '1' && (B % 2 == 0))
                B += 1;
            m++;// перебираем следующий битик
            // копируем преобразованный цвет
            newColor = Color.FromArgb(R, G, B);
            ContainerBMP.SetPixel(j, i, newColor);

            // смещаемся по сообщению
            if (m == 12)
            {
                m = 0;
                n += 1;
                bitch = "";
                if (n == MESSAGE.Length)
                {
                    flag = true;
                    break;
                }
                else
                    bitch = DecToBin(((int)MESSAGE[n]), 12);
            }
            j++;


        }
        j = 0;
        if (flag)
            break;
    }
    DataBitmap = ContainerBMP;
    return HideRezult.hrOk;

}
```

```
}
/// <summary>
///
/// </summary>
/// <param name="ContainerBMP"></param>
/// <param name="messageString"></param>
/// <param name="keyStart"></param>
/// <param name="keyEnd"></param>
/// <returns></returns>
public static HideRezult HideDataInRed(Bitmap ContainerBMP, string messageString, string keyStart, string keyEnd)
{
    int i, j;
    //Общее сообщение вместе с ключами
    string MESSAGE = keyStart + messageString + keyEnd;
    //Количество бит которые можно заменить
    int total = Convert.ToInt32((ContainerBMP.Width * ContainerBMP.Height) / 3);

    if ((MESSAGE.Length * 12) > (total * 3)) //Проверка, хватит ли нам картинки
    {
        DataBitmap = null; // Если не хватает картинки то возвращаем соотв. уведомление
        return HideRezult.hrToSmallPic;
    }
    else  // иначе если картинки хватает то
    {
        // Создаём изображение в котором будут хранится данные
        DataBitmap = new Bitmap(ContainerBMP.Width, ContainerBMP.Height);

        // Переменная которая будет давать нам знать заканчивать скрытие или нет
        bool flag = false;


        int n = 0, m = 0;// Позыции текущего бита сообщения

        string bitch;// Переменная которая содержит символ в двоичном виде
        Color col; // Структура цвета которую
        Color newColor;// Изменённая цветовая структура
        // Скрываем
        bitch = DecToBin(((int)MESSAGE[0]), 12);// Преобразуем первый символ

        j = 0;// Устанавливаем позицию X-а

        int R, G, B;// Переменные содержащие цвета
        for (i = 0; i < ContainerBMP.Height; i++)
        {
            while (j < ContainerBMP.Width)
            {
                col = ContainerBMP.GetPixel(j, i); // Считываем цветовую структуру по заданным координатам
                R = col.R;
                G = col.G;
                B = col.B;
                // Преобразуем КРАСНЫЙ цвет
                if (bitch[m] == '0' && (R % 2 != 0))
                    R -= 1;
                if (bitch[m] == '1' && (R % 2 == 0))
                    R += 1;
                m++;// перебираем следующий битик

                // копируем преобразованный цвет
                newColor = Color.FromArgb(R, G, B);
                ContainerBMP.SetPixel(j, i, newColor);

                // смещаемся по сообщению
                if (m == 12)
                {
                    m = 0;
                    n += 1;
                    bitch = "";
                    if (n == MESSAGE.Length)
                    {
                        flag = true;
                        break;
                    }
                    else
                        bitch = DecToBin(((int)MESSAGE[n]), 12);
                }
                j++;


            }
```

```
            j = 0;
            if (flag)
                break;
        }
        DataBitmap = ContainerBMP;
        return HideRezult.hrOk;

    }
}
/// <summary>
///
/// </summary>
/// <param name="ContainerBMP"></param>
/// <param name="messageString"></param>
/// <param name="keyStart"></param>
/// <param name="keyEnd"></param>
/// <param name="pos_X"></param>
/// <param name="pos_Y"></param>
/// <returns></returns>
public static HideRezult HideDataInRed(Bitmap ContainerBMP, string messageString, string keyStart, string keyEnd, int pos_X, int pos_Y)
{
    int i, j;
    //Общее сообщение вместе с ключами
    string MESSAGE = keyStart + messageString + keyEnd;
    //Количество бит которые можно заменить
    int total = Convert.ToInt32((ContainerBMP.Width * ContainerBMP.Height - (pos_Y * ContainerBMP.Width + pos_X)) / 3);
    if ((MESSAGE.Length * 12) > (total * 3)) //Проверка, хватит ли нам картинки
    {
        DataBitmap = null; // Если не хватает картинки то возвращаем соотв. уведомление
        return HideRezult.hrToSmallPic;
    }
    else  // иначе если картинки хватает то
    {
        // Создаём изображение в котором будут хранится данные
        DataBitmap = new Bitmap(ContainerBMP.Width, ContainerBMP.Height);

        // Переменная которая будет давать нам знать заканчивать скрытие или нет
        bool flag = false;


        int n = 0, m = 0;// Позыции текущего бита сообщения

        string bitch;// Переменная которая содержит символ в двоичном виде
        Color col; // Структура цвета которую
        Color newColor;// Изменённая цветовая структура
        // Скрываем
        bitch = DecToBin(((int)MESSAGE[0]), 12);// Преобразуем первый символ

        j = pos_X;// Устанавливаем позицию X-а

        int R, G, B;// Переменные содержащие цвета
        for (i = pos_Y; i < ContainerBMP.Height; i++)
        {
            while (j < ContainerBMP.Width)
            {
                col = ContainerBMP.GetPixel(j, i); // Считываем цветовую структуру по заданным координатам
                R = col.R;
                G = col.G;
                B = col.B;
                // Преобразуем КРАСНЫЙ цвет
                if (bitch[m] == '0' && (R % 2 != 0))
                    R -= 1;
                if (bitch[m] == '1' && (R % 2 == 0))
                    R += 1;
                m++;// перебираем следующий битик

                // копируем преобразованный цвет
                newColor = Color.FromArgb(R, G, B);
                ContainerBMP.SetPixel(j, i, newColor);

                // смещаемся по сообщению
                if (m == 12)
                {
                    m = 0;
                    n += 1;
                    bitch = "";
                    if (n == MESSAGE.Length)
                    {
                        flag = true;
```

```
                    break;
                }
                else
                    bitch = DecToBin(((int)MESSAGE[n]), 12);
            }
            j++;


        }
        j = 0;
        if (flag)
            break;
    }
    DataBitmap = ContainerBMP;
    return HideRezult.hrOk;

  }
}

/// <summary>
///
/// </summary>
/// <param name="ContainerBMP"></param>
/// <param name="messageString"></param>
/// <param name="keyStart"></param>
/// <param name="keyEnd"></param>
/// <returns></returns>
public static HideRezult HideDataInGreen(Bitmap ContainerBMP, string messageString, string keyStart, string keyEnd)
{
  int i, j;
  //Общее сообщение вместе с ключами
  string MESSAGE = keyStart + messageString + keyEnd;
  //Количество бит которые можно заменить
  int total = Convert.ToInt32((ContainerBMP.Width * ContainerBMP.Height) / 3);

  if ((MESSAGE.Length * 12) > (total * 3)) //Проверка, хватит ли нам картинки
  {
    DataBitmap = null; // Если не хватает картинки то возвращаем соотв. уведомление
    return HideRezult.hrToSmallPic;
  }
  else  // иначе если картинки хватает то
  {
    // Создаём изображение в котором будут хранится данные
    DataBitmap = new Bitmap(ContainerBMP.Width, ContainerBMP.Height);

    // Переменная которая будет давать нам знать заканчивать скрытие или нет
    bool flag = false;


    int n = 0, m = 0;// Позыции текущего бита сообщения

    string bitch;// Переменная которая содержит символ в двоичном виде
    Color col; // Структура цвета которую
    Color newColor;// Изменённая цветовая структура
    // Скрываем
    bitch = DecToBin(((int)MESSAGE[0]), 12);// Преобразуем первый символ

    j = 0;// Устанавливаем позицию X-а

    int R, G, B;// Переменные содержащие цвета
    for (i = 0; i < ContainerBMP.Height; i++)
    {
      while (j < ContainerBMP.Width)
      {
        col = ContainerBMP.GetPixel(j, i); // Считываем цветовую структуру по заданным координатам
        R = col.R;
        G = col.G;
        B = col.B;
        // Преобразуем ЗЕЛЁНЫЙ цвет
        if (bitch[m] == '0' && (G % 2 != 0))
          G -= 1;
        if (bitch[m] == '1' && (G % 2 == 0))
          G += 1;
        m++;// перебираем следующий битик

        // копируем преобразованный цвет
        newColor = Color.FromArgb(R, G, B);
        ContainerBMP.SetPixel(j, i, newColor);
```

```
                // смещаемся по сообщению
                if (m == 12)
                {
                    m = 0;
                    n += 1;
                    bitch = "";
                    if (n == MESSAGE.Length)
                    {
                        flag = true;
                        break;
                    }
                    else
                        bitch = DecToBin(((int)MESSAGE[n]), 12);
                }
                j++;


            }
            j = 0;
            if (flag)
                break;
        }
        DataBitmap = ContainerBMP;
        return HideRezult.hrOk;

    }
}
/// <summary>
///
/// </summary>
/// <param name="ContainerBMP"></param>
/// <param name="messageString"></param>
/// <param name="keyStart"></param>
/// <param name="keyEnd"></param>
/// <param name="pos_X"></param>
/// <param name="pos_Y"></param>
/// <returns></returns>
public static HideRezult HideDataInGreen(Bitmap ContainerBMP, string messageString, string keyStart, string keyEnd, int pos_X, int pos_Y)
{
    int i, j;
    //Общее сообщение вместе с ключами
    string MESSAGE = keyStart + messageString + keyEnd;
    //Количество бит которые можно заменить
    int total = Convert.ToInt32((ContainerBMP.Width * ContainerBMP.Height - (pos_Y * ContainerBMP.Width + pos_X)) / 3);
    if ((MESSAGE.Length * 12) > (total * 3)) //Проверка, хватит ли нам картинки
    {
        DataBitmap = null; // Если не хватает картинки то возвращаем соотв. уведомление
        return HideRezult.hrToSmallPic;
    }
    else  // иначе если картинки хватает то
    {
        // Создаём изображение в котором будут хранится данные
        DataBitmap = new Bitmap(ContainerBMP.Width, ContainerBMP.Height);

        // Переменная которая будет давать нам знать заканчивать скрытие или нет
        bool flag = false;


        int n = 0, m = 0;// Позыции текущего бита сообщения

        string bitch;// Переменная которая содержит символ в двоичном виде
        Color col; // Структура цвета которую
        Color newColor;// Изменённая цветовая структура
        // Скрываем
        bitch = DecToBin(((int)MESSAGE[0]), 12);// Преобразуем первый символ

        j = pos_X;// Устанавливаем позицию X-а

        int R, G, B;// Переменные содержащие цвета
        for (i = pos_Y; i < ContainerBMP.Height; i++)
        {
            while (j < ContainerBMP.Width)
            {
                col = ContainerBMP.GetPixel(j, i); // Считываем цветовую структуру по заданным координатам
                R = col.R;
                G = col.G;
                B = col.B;
                // Преобразуем ЗЕЛЁНЫЙ цвет
                if (bitch[m] == '0' && (G % 2 != 0))
```

```
                G -= 1;
            if (bitch[m] == '1' && (G % 2 == 0))
                G += 1;
            m++;// перебираем следующий битик

            // копируем преобразованный цвет
            newColor = Color.FromArgb(R, G, B);
            ContainerBMP.SetPixel(j, i, newColor);

            // смещаемся по сообщению
            if (m == 12)
            {
                m = 0;
                n += 1;
                bitch = "";
                if (n == MESSAGE.Length)
                {
                    flag = true;
                    break;
                }
                else
                    bitch = DecToBin(((int)MESSAGE[n]), 12);
            }
            j++;


        }
        j = 0;
        if (flag)
            break;
    }
    DataBitmap = ContainerBMP;
    return HideRezult.hrOk;

  }
}

/// <summary>
///
/// </summary>
/// <param name="ContainerBMP"></param>
/// <param name="messageString"></param>
/// <param name="keyStart"></param>
/// <param name="keyEnd"></param>
/// <returns></returns>
public static HideRezult HideDataInBlue(Bitmap ContainerBMP, string messageString, string keyStart, string keyEnd)
{
  int i, j;
  //Общее сообщение вместе с ключами
  string MESSAGE = keyStart + messageString + keyEnd;
  //Количество бит которые можно заменить
  int total = Convert.ToInt32((ContainerBMP.Width * ContainerBMP.Height) / 3);

  if ((MESSAGE.Length * 12) > (total * 3)) //Проверка, хватит ли нам картинки
  {
    DataBitmap = null; // Если не хватает картинки то возвращаем соотв. уведомление
    return HideRezult.hrToSmallPic;
  }
  else  // иначе если картинки хватает то
  {
    // Создаём изображение в котором будут хранится данные
    DataBitmap = new Bitmap(ContainerBMP.Width, ContainerBMP.Height);

    // Переменная которая будет давать нам знать заканчивать скрытие или нет
    bool flag = false;


    int n = 0, m = 0;// Позыции текущего бита сообщения

    string bitch;// Переменная которая содержит символ в двоичном вииде
    Color col; // Структура цвета которую
    Color newColor;// Изменённая цветовая структура
    // Скрываем
    bitch = DecToBin(((int)MESSAGE[0]), 12);// Преобразуем первый символ

    j = 0;// Устанавливаем позицию X-а

    int R, G, B;// Переменные содержащие цвета
    for (i = 0; i < ContainerBMP.Height; i++)
```

```
        {
            while (j < ContainerBMP.Width)
            {
                col = ContainerBMP.GetPixel(j, i); // Считываем цветовую структуру по заданным координатам
                R = col.R;
                G = col.G;
                B = col.B;
                // Преобразуем СИНИЙ цвет
                if (bitch[m] == '0' && (B % 2 != 0))
                    B -= 1;
                if (bitch[m] == '1' && (B % 2 == 0))
                    B += 1;
                m++;// перебираем следующий битик

                // копируем преобразованный цвет
                newColor = Color.FromArgb(R, G, B);
                ContainerBMP.SetPixel(j, i, newColor);

                // смещаемся по сообщению
                if (m == 12)
                {
                    m = 0;
                    n += 1;
                    bitch = "";
                    if (n == MESSAGE.Length)
                    {
                        flag = true;
                        break;
                    }
                    else
                        bitch = DecToBin(((int)MESSAGE[n]), 12);
                }
                j++;


            }
            j = 0;
            if (flag)
                break;
        }
        DataBitmap = ContainerBMP;
        return HideRezult.hrOk;

    }
}
/// <summary>
///
/// </summary>
/// <param name="ContainerBMP"></param>
/// <param name="messageString"></param>
/// <param name="keyStart"></param>
/// <param name="keyEnd"></param>
/// <param name="pos_X"></param>
/// <param name="pos_Y"></param>
/// <returns></returns>
public static HideRezult HideDataInBlue(Bitmap ContainerBMP, string messageString, string keyStart, string keyEnd, int pos_X, int pos_Y)
{
    int i, j;
    //Общее сообщение вместе с ключами
    string MESSAGE = keyStart + messageString + keyEnd;
    //Количество бит которые можно заменить
    int total = Convert.ToInt32((ContainerBMP.Width * ContainerBMP.Height - (pos_Y * ContainerBMP.Width + pos_X)) / 3);

    if ((MESSAGE.Length * 12) > (total * 3)) //Проверка, хватит ли нам картинки
    {
        DataBitmap = null; // Если не хватает картинки то возвращаем соотв. уведомление
        return HideRezult.hrToSmallPic;
    }
    else  // иначе если картинки хватает то
    {
        // Создаём изображение в котором будут хранится данные
        DataBitmap = new Bitmap(ContainerBMP.Width, ContainerBMP.Height);

        // Переменная которая будет давать нам знать заканчивать скрытие или нет
        bool flag = false;


        int n = 0, m = 0;// Позыции текущего бита сообщения
```

```
        string bitch;// Переменная которая содержит символ в двоичном виде
        Color col; // Структура цвета которую
        Color newColor;// Изменённая цветовая структура
        // Скрываем
        bitch = DecToBin(((int)MESSAGE[0]), 12);// Преобразуем первый символ

        j = pos_X;// Устанавливаем позицию X-а

        int R, G, B;// Переменные содержащие цвета
        for (i = pos_Y; i < ContainerBMP.Height; i++)
        {
            while (j < ContainerBMP.Width)
            {
                col = ContainerBMP.GetPixel(j, i); // Считываем цветовую структуру по заданным координатам
                R = col.R;
                G = col.G;
                B = col.B;
                // Преобразуем СИНИЙ цвет
                if (bitch[m] == '0' && (B % 2 != 0))
                    B -= 1;
                if (bitch[m] == '1' && (B % 2 == 0))
                    B += 1;
                m++;// перебираем следующий битик

                // копируем преобразованный цвет
                newColor = Color.FromArgb(R, G, B);
                ContainerBMP.SetPixel(j, i, newColor);

                // смещаемся по сообщению
                if (m == 12)
                {
                    m = 0;
                    n += 1;
                    bitch = "";
                    if (n == MESSAGE.Length)
                    {
                        flag = true;
                        break;
                    }
                    else
                        bitch = DecToBin(((int)MESSAGE[n]), 12);
                }
                j++;


            }
            j = 0;
            if (flag)
                break;
        }
        DataBitmap = ContainerBMP;
        return HideRezult.hrOk;

    }
}

/// <summary>
/// Функция извлекает сообщение, расположенное между двумя ключами, из картинки
/// </summary>
/// <param name="DataBMP">Картинка в формате BMP со скрытыми данными</param>
/// <param name="keyStart"></param>
/// <param name="keyEnd"></param>
/// <returns>Возвращает извлечённое сообщение</returns>
public static string ExtractData(Bitmap DataBMP, string keyStart, string keyEnd)
{
    string extractMsg=""; // Извлечённое сообщение
    string extractBit=""; // Извлечённый бит сообщения
    bool flag = false; // Переменная указывающая на то нужно ли нам продолжать извлечение или нет
    Color col; // Цветовая структра которую мы считываем с пиксела по заданным координатам

    int i, j;
    j = 0;
    for (i = 0; i < DataBMP.Height; i++)
    {
        while (j < DataBMP.Width)
        {
            col = DataBMP.GetPixel(j, i);
            if (col.R % 2 != 0)
                extractBit += "1";
```

```
            if (col.R % 2 == 0)
                extractBit += "0";
            if (col.G % 2 != 0)
                extractBit += "1";
            if (col.G % 2 == 0)
                extractBit += "0";
            if (col.B % 2 != 0)
                extractBit += "1";
            if (col.B % 2 == 0)
                extractBit += "0";

            if (extractBit.Length == 12)
            {
                extractMsg += Convert.ToString((char)BinToDec(extractBit));
                extractBit = "";
                if (extractMsg.IndexOf(keyEnd) != -1)
                {
                    flag = true;
                    break;
                }

            }
            j++;

        }
        j = 0;
        if (flag)
            break;

    }
    if (extractMsg.IndexOf(keyStart) != -1 && extractMsg.IndexOf(keyEnd) != -1)
    {
        string tempSTR;
        int posStart, posEnd;

        posStart = extractMsg.IndexOf(keyStart);
        posEnd = extractMsg.IndexOf(keyEnd);

        if (posEnd > posStart)
        {
            tempSTR = extractMsg.Remove(0, keyStart.Length);
            extractMsg = tempSTR.Remove(tempSTR.Length - keyEnd.Length, keyEnd.Length);
        }
        else
            return "Что-то не впорядке с ключами!!!";

        return extractMsg;

    }
    else
        return "Неверные ключи!!!";



}
/// <summary>
///
/// </summary>
/// <param name="DataBMP"></param>
/// <param name="keyStart"></param>
/// <param name="keyEnd"></param>
/// <param name="pos_X"></param>
/// <param name="pos_Y"></param>
/// <returns></returns>
public static string ExtractData(Bitmap DataBMP, string keyStart, string keyEnd, int pos_X, int pos_Y)
{
    string extractMsg = ""; // Извлечённое сообщение
    string extractBit = ""; // Извлечённый бит сообщения
    bool flag = false; // Переменная указывающая на то нужно ли нам продолжать извлечение или нет
    Color col; // Цветовая структра которую мы считываем с пиксела по заданным координатам

    int i, j;
    j = pos_X;
    for (i = pos_Y; i < DataBMP.Height; i++)
    {
        while (j < DataBMP.Width)
        {
            col = DataBMP.GetPixel(j, i);
```

```
            if (col.R % 2 != 0)
                extractBit += "1";
            if (col.R % 2 == 0)
                extractBit += "0";
            if (col.G % 2 != 0)
                extractBit += "1";
            if (col.G % 2 == 0)
                extractBit += "0";
            if (col.B % 2 != 0)
                extractBit += "1";
            if (col.B % 2 == 0)
                extractBit += "0";

            if (extractBit.Length == 12)
            {
                extractMsg += Convert.ToString((char)BinToDec(extractBit));
                extractBit = "";
                if (extractMsg.IndexOf(keyEnd) != -1)
                {
                    flag = true;
                    break;
                }

            }
            j++;

        }
        j = 0;
        if (flag)
            break;

    }
    if (extractMsg.IndexOf(keyStart) != -1 && extractMsg.IndexOf(keyEnd) != -1)
    {
        string tempSTR;
        int posStart, posEnd;

        posStart = extractMsg.IndexOf(keyStart);
        posEnd = extractMsg.IndexOf(keyEnd);

        if (posEnd > posStart)
        {
            tempSTR = extractMsg.Remove(0, keyStart.Length);
            extractMsg = tempSTR.Remove(tempSTR.Length - keyEnd.Length, keyEnd.Length);
        }
        else
            return "Что-то не впорядке с ключами!!!";

        return extractMsg;

    }
    else
        return "Неверные ключи!!!";

}

/// <summary>
/// Функция извлекает сообщение, расположенное между двумя ключами, из картинки
/// </summary>
/// <param name="DataBMP">Картинка в формате BMP со скрытыми данными</param>
/// <param name="keyStart"></param>
/// <param name="keyEnd"></param>
/// <returns>Возвращает извлечённое сообщение</returns>
public static string ExtractDataFromRed(Bitmap DataBMP, string keyStart, string keyEnd)
{
    string extractMsg = ""; // Извлечённое сообщение
    string extractBit = ""; // Извлечённый бит сообщения
    bool flag = false; // Переменная указывающая на то нужно ли нам продолжать извлечение или нет
    Color col; // Цветовая структра которую мы считываем с пиксела по заданным координатам

    int i, j;
    j = 0;
    for (i = 0; i < DataBMP.Height; i++)
    {
        while (j < DataBMP.Width)
        {
            col = DataBMP.GetPixel(j, i);
            if (col.R % 2 != 0)
                extractBit += "1";
```

```
            if (col.R % 2 == 0)
                extractBit += "0";
            //
            if (extractBit.Length == 12)
            {
                extractMsg += Convert.ToString((char)BinToDec(extractBit));
                extractBit = "";
                if (extractMsg.IndexOf(keyEnd) != -1)
                {
                    flag = true;
                    break;
                }

            }
            j++;

        }
        j = 0;
        if (flag)
            break;

    }
    if (extractMsg.IndexOf(keyStart) != -1 && extractMsg.IndexOf(keyEnd) != -1)
    {
        string tempSTR;
        int posStart, posEnd;

        posStart = extractMsg.IndexOf(keyStart);
        posEnd = extractMsg.IndexOf(keyEnd);

        if (posEnd > posStart)
        {
            tempSTR = extractMsg.Remove(0, keyStart.Length);
            extractMsg = tempSTR.Remove(tempSTR.Length - keyEnd.Length, keyEnd.Length);
        }
        else
            return "Что-то не впорядке с ключами!!!";

        return extractMsg;

    }
    else
        return "Неверные ключи!!!";



}
/// <summary>
///
/// </summary>
/// <param name="DataBMP"></param>
/// <param name="keyStart"></param>
/// <param name="keyEnd"></param>
/// <param name="pos_X"></param>
/// <param name="pos_Y"></param>
/// <returns></returns>
public static string ExtractDataFromRed(Bitmap DataBMP, string keyStart, string keyEnd, int pos_X, int pos_Y)
{
    string extractMsg = ""; // Извлечённое сообщение
    string extractBit = ""; // Извлечённый бит сообщения
    bool flag = false; // Переменная указывающая на то нужно ли нам продолжать извлечение или нет
    Color col; // Цветовая структра которую мы считываем с пиксела по заданным координатам

    int i, j;
    j = pos_X;
    for (i = pos_Y; i < DataBMP.Height; i++)
    {
        while (j < DataBMP.Width)
        {
            col = DataBMP.GetPixel(j, i);
            if (col.R % 2 != 0)
                extractBit += "1";
            if (col.R % 2 == 0)
                extractBit += "0";
            //
            if (extractBit.Length == 12)
            {
                extractMsg += Convert.ToString((char)BinToDec(extractBit));
```

```
                extractBit = "";
                if (extractMsg.IndexOf(keyEnd) != -1)
                {
                    flag = true;
                    break;
                }

            }
            j++;

        }
        j = 0;
        if (flag)
            break;

    }
    if (extractMsg.IndexOf(keyStart) != -1 && extractMsg.IndexOf(keyEnd) != -1)
    {
        string tempSTR;
        int posStart, posEnd;

        posStart = extractMsg.IndexOf(keyStart);
        posEnd = extractMsg.IndexOf(keyEnd);

        if (posEnd > posStart)
        {
            tempSTR = extractMsg.Remove(0, keyStart.Length);
            extractMsg = tempSTR.Remove(tempSTR.Length - keyEnd.Length, keyEnd.Length);
        }
        else
            return "Что-то не впорядке с ключами!!!";

        return extractMsg;

    }
    else
        return "Неверные ключи!!!";

}

/// <summary>
/// Функция извлекает сообщение, расположенное между двумя ключами, из картинки
/// </summary>
/// <param name="DataBMP">Картинка в формате BMP со скрытыми данными</param>
/// <param name="keyStart"></param>
/// <param name="keyEnd"></param>
/// <returns>Возвращает извлечённое сообщение</returns>
public static string ExtractDataFromGreen(Bitmap DataBMP, string keyStart, string keyEnd)
{
    string extractMsg = ""; // Извлечённое сообщение
    string extractBit = ""; // Извлечённый бит сообщения
    bool flag = false; // Переменная указывающая на то нужно ли нам продолжать извлечение или нет
    Color col; // Цветовая структра которую мы считываем с пиксела по заданным координатам

    int i, j;
    j = 0;
    for (i = 0; i < DataBMP.Height; i++)
    {
        while (j < DataBMP.Width)
        {
            col = DataBMP.GetPixel(j, i);

            if (col.G % 2 != 0)
                extractBit += "1";
            if (col.G % 2 == 0)
                extractBit += "0";

            if (extractBit.Length == 12)
            {
                extractMsg += Convert.ToString((char)BinToDec(extractBit));
                extractBit = "";
                if (extractMsg.IndexOf(keyEnd) != -1)
                {
                    flag = true;
                    break;
                }

            }
            j++;
```

```
            }
            j = 0;
            if (flag)
                break;

        }
        if (extractMsg.IndexOf(keyStart) != -1 && extractMsg.IndexOf(keyEnd) != -1)
        {
            string tempSTR;
            int posStart, posEnd;

            posStart = extractMsg.IndexOf(keyStart);
            posEnd = extractMsg.IndexOf(keyEnd);

            if (posEnd > posStart)
            {
                tempSTR = extractMsg.Remove(0, keyStart.Length);
                extractMsg = tempSTR.Remove(tempSTR.Length - keyEnd.Length, keyEnd.Length);
            }
            else
                return "Что-то не впорядке с ключами!!!";

            return extractMsg;

        }
        else
            return "Неверные ключи!!!";



}
/// <summary>
///
/// </summary>
/// <param name="DataBMP"></param>
/// <param name="keyStart"></param>
/// <param name="keyEnd"></param>
/// <param name="pos_X"></param>
/// <param name="pos_Y"></param>
/// <returns></returns>
public static string ExtractDataFromGreen(Bitmap DataBMP, string keyStart, string keyEnd, int pos_X, int pos_Y)
{
    string extractMsg = ""; // Извлечённое сообщение
    string extractBit = ""; // Извлечённый бит сообщения
    bool flag = false; // Переменная указывающая на то нужно ли нам продолжать извлечение или нет
    Color col; // Цветовая структра которую мы считываем с пиксела по заданным координатам

    int i, j;
    j = pos_X;
    for (i = pos_Y; i < DataBMP.Height; i++)
    {
        while (j < DataBMP.Width)
        {
            col = DataBMP.GetPixel(j, i);

            if (col.G % 2 != 0)
                extractBit += "1";
            if (col.G % 2 == 0)
                extractBit += "0";


            if (extractBit.Length == 12)
            {
                extractMsg += Convert.ToString((char)BinToDec(extractBit));
                extractBit = "";
                if (extractMsg.IndexOf(keyEnd) != -1)
                {
                    flag = true;
                    break;
                }

            }
            j++;

        }
        j = 0;
        if (flag)
```

```
            break;

        }
        if (extractMsg.IndexOf(keyStart) != -1 && extractMsg.IndexOf(keyEnd) != -1)
        {
            string tempSTR;
            int posStart, posEnd;

            posStart = extractMsg.IndexOf(keyStart);
            posEnd = extractMsg.IndexOf(keyEnd);

            if (posEnd > posStart)
            {
                tempSTR = extractMsg.Remove(0, keyStart.Length);
                extractMsg = tempSTR.Remove(tempSTR.Length - keyEnd.Length, keyEnd.Length);
            }
            else
                return "Что-то не впорядке с ключами!!!";

            return extractMsg;

        }
        else
            return "Неверные ключи!!!";

    }

    /// <summary>
    /// Функция извлекает сообщение, расположенное между двумя ключами, из картинки
    /// </summary>
    /// <param name="DataBMP">Картинка в формате BMP со скрытыми данными</param>
    /// <param name="keyStart"></param>
    /// <param name="keyEnd"></param>
    /// <returns>Возвращает извлечённое сообщение</returns>
    public static string ExtractDataFromBlue(Bitmap DataBMP, string keyStart, string keyEnd)
    {
        string extractMsg = ""; // Извлечённое сообщение
        string extractBit = ""; // Извлечённый бит сообщения
        bool flag = false; // Переменная указывающая на то нужно ли нам продолжать извлечение или нет
        Color col; // Цветовая структра которую мы считываем с пиксела по заданным координатам

        int i, j;
        j = 0;
        for (i = 0; i < DataBMP.Height; i++)
        {
            while (j < DataBMP.Width)
            {
                col = DataBMP.GetPixel(j, i);

                if (col.B % 2 != 0)
                    extractBit += "1";
                if (col.B % 2 == 0)
                    extractBit += "0";

                if (extractBit.Length == 12)
                {
                    extractMsg += Convert.ToString((char)BinToDec(extractBit));
                    extractBit = "";
                    if (extractMsg.IndexOf(keyEnd) != -1)
                    {
                        flag = true;
                        break;
                    }

                }
                j++;

            }
            j = 0;
            if (flag)
                break;

        }
        if (extractMsg.IndexOf(keyStart) != -1 && extractMsg.IndexOf(keyEnd) != -1)
        {
            string tempSTR;
            int posStart, posEnd;

            posStart = extractMsg.IndexOf(keyStart);
```

```
            posEnd = extractMsg.IndexOf(keyEnd);

            if (posEnd > posStart)
            {
                tempSTR = extractMsg.Remove(0, keyStart.Length);
                extractMsg = tempSTR.Remove(tempSTR.Length - keyEnd.Length, keyEnd.Length);
            }
            else
                return "Что-то не впорядке с ключами!!!";

            return extractMsg;

        }
        else
            return "Неверные ключи!!!";



    }
    /// <summary>
    ///
    /// </summary>
    /// <param name="DataBMP"></param>
    /// <param name="keyStart"></param>
    /// <param name="keyEnd"></param>
    /// <param name="pos_X"></param>
    /// <param name="pos_Y"></param>
    /// <returns></returns>
    public static string ExtractDataFromBlue(Bitmap DataBMP, string keyStart, string keyEnd, int pos_X, int pos_Y)
    {
        string extractMsg = ""; // Извлечённое сообщение
        string extractBit = ""; // Извлечённый бит сообщения
        bool flag = false; // Переменная указывающая на то нужно ли нам продолжать извлечение или нет
        Color col; // Цветовая структра которую мы считываем с пиксела по заданным координатам

        int i, j;
        j = pos_X;
        for (i = pos_Y; i < DataBMP.Height; i++)
        {
            while (j < DataBMP.Width)
            {
                col = DataBMP.GetPixel(j, i);

                if (col.B % 2 != 0)
                    extractBit += "1";
                if (col.B % 2 == 0)
                    extractBit += "0";


                if (extractBit.Length == 12)
                {
                    extractMsg += Convert.ToString((char)BinToDec(extractBit));
                    extractBit = "";
                    if (extractMsg.IndexOf(keyEnd) != -1)
                    {
                        flag = true;
                        break;
                    }

                }
                j++;

            }
            j = 0;
            if (flag)
                break;

        }
        if (extractMsg.IndexOf(keyStart) != -1 && extractMsg.IndexOf(keyEnd) != -1)
        {
            string tempSTR;
            int posStart, posEnd;

            posStart = extractMsg.IndexOf(keyStart);
            posEnd = extractMsg.IndexOf(keyEnd);

            if (posEnd > posStart)
            {
```

```
                tempSTR = extractMsg.Remove(0, keyStart.Length);
                extractMsg = tempSTR.Remove(tempSTR.Length - keyEnd.Length, keyEnd.Length);
            }
            else
                return "Что-то не впорядке с ключами!!!";

            return extractMsg;

        }
        else
            return "Неверные ключи!!!";

    }
  }
}
```

ДОДАТОК В

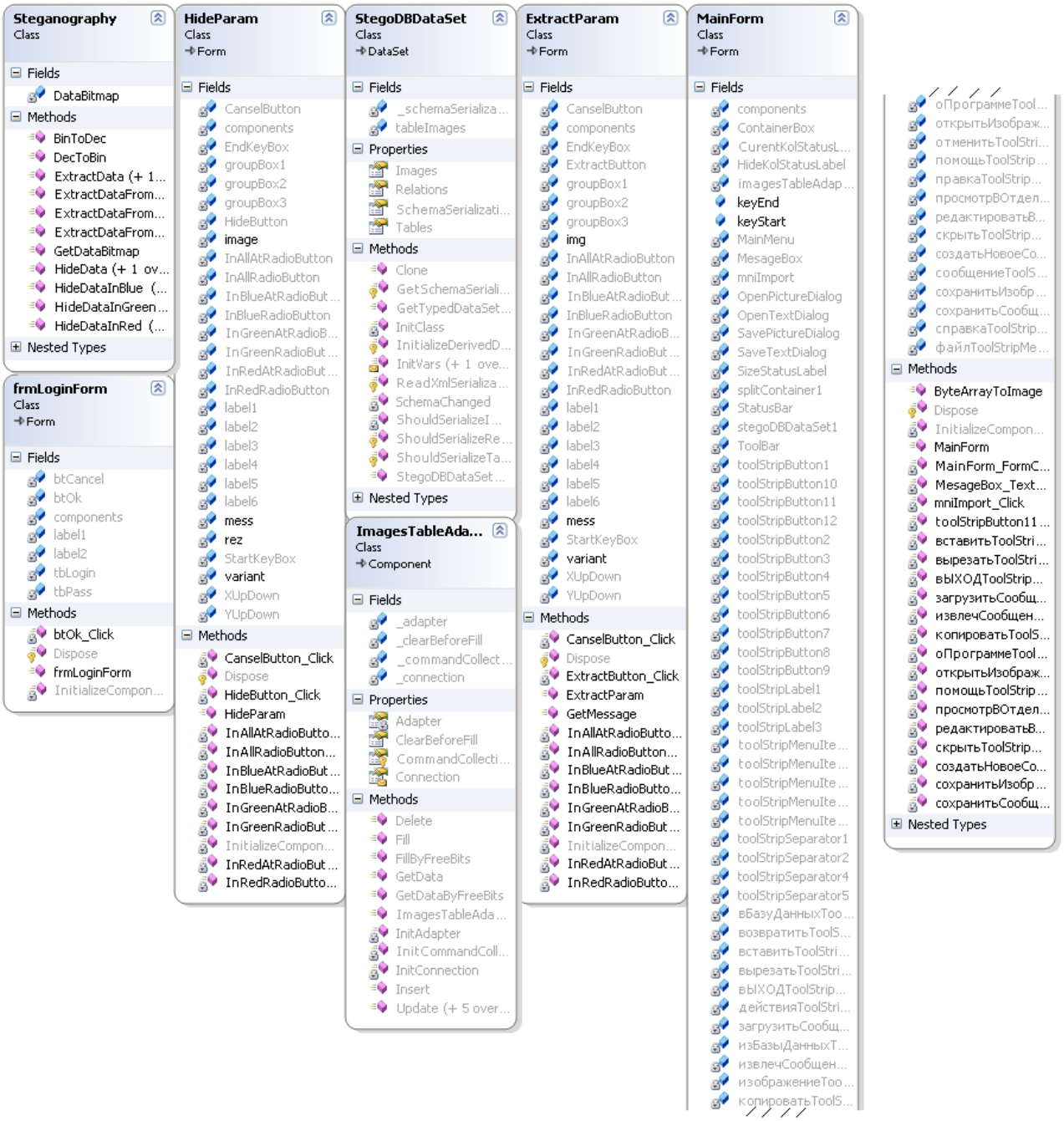
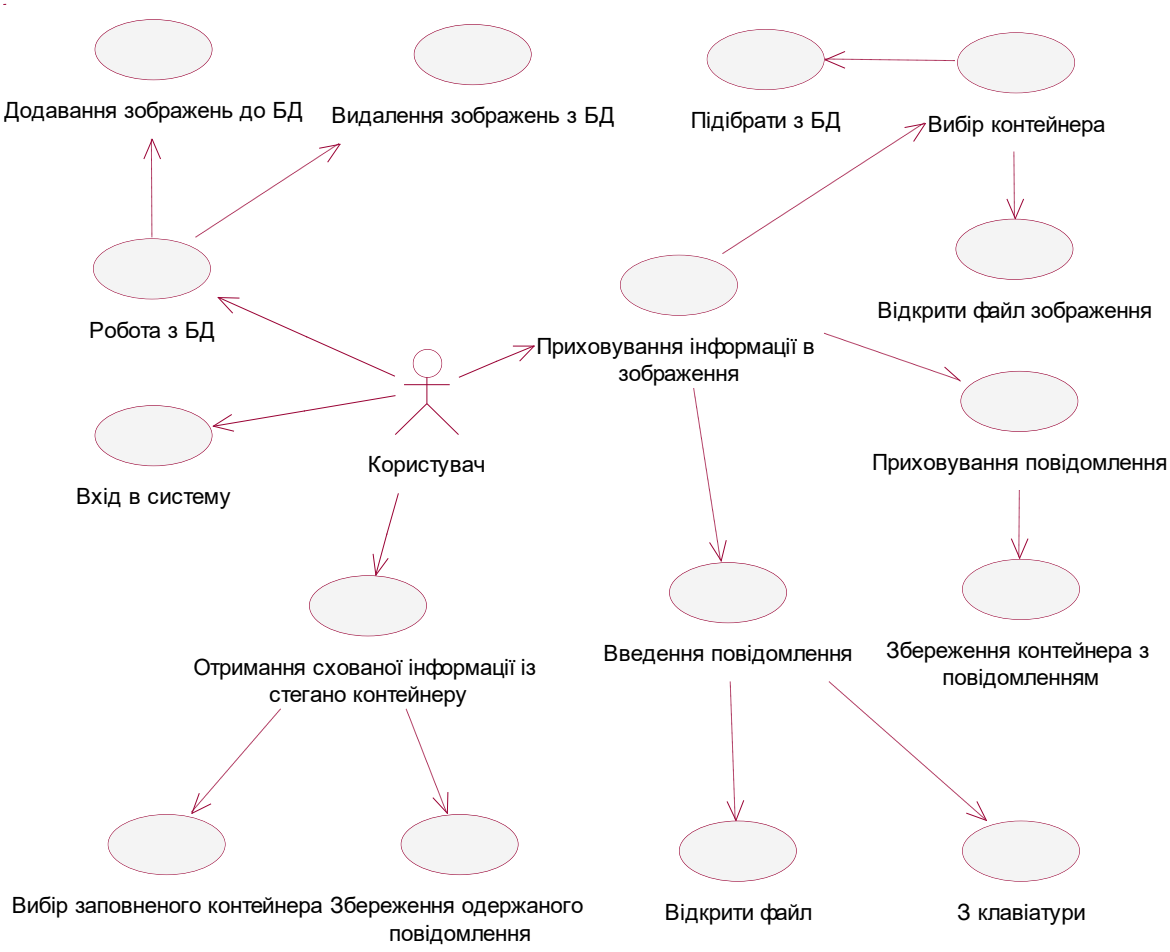


Рис В.1. Діаграмма класів.

ДОДАТОК Д

ДІАГРАМА ВИКОРИСТАННЯ



ДОДАТОК Е

Таблиця Е.1

Тестові сценарії авторизації користувача у системі

№ з/п	Крок сценарію	Очікуваний результат	Отриманий результат	Відмітка про проходження кроку сценарію (Так/Ні)
1	Запустити додаток	Повинне з’явитися вікно авторизації	Вікно авторизації з’являється	Так
2	Перевірка чи реєстрований користувач у системі тобто перевірка логіна та пароля	У разі вводу у поля логін та пароль даних не існуючого користувача повинно з’явитися повідомлення: «Неверный логин или пароль»	повідомлення: «Неверный логин или»з’являється	Так
3	Увести ім’я користувача Admin	Повинне з’явитися повідомлення: «Необходимо заполнить все поля»	Повідомлення: «Необходимо заполнить все поля» з’являється	Так
4	Увести пароль користувача 1111111	Повинне з’явитися повідомлення: «Неверный логин или пароль или такой пользователь не зарегистрирован»	Повідомлення: «Неверный логин или пароль или такой пользователь не зарегистрирован» з’являється	Так
5	Увести пароль користувача adminadmin	Повинне відкритися головне вікно програми	головне вікно програми з’явилося	Так

Відмітка про проходження тесту (пройдений/не пройдений): пройдений