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

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“____” _____ 2025 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за першим (бакалаврським) рівнем вищої освіти
на тему
“Програмне забезпечення системи відеонагляду для IoT”

КбПЗ_2025

Виконав здобувач вищої освіти
IV курсу, групи КІ-21-2
ОПП «Комп’ютерна інженерія»
спеціальності 123 «Комп’ютерна інженерія»
_____ Полюхович І.В.
«____» _____ 2025 р.

Керівник проекту
кандидат технічних наук, доцент
_____ Коваленко А.С.
«____» _____ 2025 р.
Рецензент _____
_____

м. Кропивницький

Центральноукраїнський національний технічний університет

Факультет *Механіко-технологічний*

Кафедра *Кібербезпеки та програмного забезпечення*

Освітній ступінь *бакалавр*

Галузь знань ._12 _____*"*Інформаційні технології*"*_____

Спеціальність *123 "*Комп'ютерна інженерія*"*

Освітньо-професійна (освітньо-наукова) програма _*"*Комп'ютерна інженерія*"*_

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____ Олексій СМІРНОВ
« 17 » січня 2025 року

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ПЕРШИМ (БАКАЛАВРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Полюховичу Іллі Віталійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи *Програмне забезпечення системи відеонагляду для IoT*

2. Керівник роботи *Коваленко Анна Степанівна, канд. техн. наук, доцент*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 47-02 від 17.01.2025 року

3. Строк подання студентом роботи до захисту *23.05.2025 р.*

4. Мета та завдання випускної кваліфікаційної роботи*: Метою роботи є розробка програмного забезпечення системи відеонагляду для IoT*

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

1. Призначення та область використання.

2.Перегляд аналогічних існуючих систем.

3. Опис і обґрунтування проектних рішень.

4. Етапи програмування системи.

5. Впровадження системи в промислову експлуатацію.

6. Висновки

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

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

КПІ 2025

7. Дата видачі завдання « 17 » січня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.03.2025 р.	
2.	Постановка задачі, оформлення ТЗ	15.03.2025 р.	
3.	Розробка моделі компонента	20.03.2025 р.	
4.	Розробка структур даних	25.03.2025 р.	
5.	Розробка алгоритмів зв’язку та відображення	30.03.2025 р.	
6.	Програмування алгоритмів	10.04.2025 р.	
7.	Оформлення ПЗ	17.04.2025 р.	
8.	Попередній захист роботи	23.05.2025 р.	

КбПЗ_2025

Дата видачі завдання
« 17 » січня 2025 р.

Підпис керівника
_____ Коваленко А.С.
(прізвище та ініціали)

Завдання прийнято до виконання
« 17 » січня 2025 р.

Підпис здобувача
_____ Полюхович І.В.
(прізвище та ініціали)

АНОТАЦІЯ

Полюхович І.В. Програмне забезпечення системи відеонагляду для IoT. 123 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2025.

В даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи відеонагляду для IoT.

Метою розробки є програмне забезпечення системи відеонагляду для IoT.

Результат роботи – програмна реалізація системи відеонагляду для IoT.

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

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

Програма може використовуватися на ПЕОМ з ОС Windows 10/11.

Програму розроблено в середовищі Builder C++.

Ключові слова: комп'ютерна інженерія, відеонагляд, IoT

КбПЗ-2025

ABSTRACT

Polyukhovich I.V. Software for a video surveillance system for IoT. 123 Computer Engineering. Central Ukrainian National Technical University. Kropyvnytskyi. 2025.

In this final qualification work for the first (bachelor's) level of higher education, software has been developed that is intended for a video surveillance system for IoT.

The purpose of the development is software for a video surveillance system for IoT.

The result of the work is a software implementation of a video surveillance system for IoT.

In the process of working on the software model, an analysis of existing hardware and software was performed. All components of the developed software are fully described.

A convenient user interface has been developed. Instructions for working with software are provided.

The program can be used on a PC with Windows 10/11.

The program was developed in the Builder C++ environment.

Keywords: computer engineering, video surveillance, IoT

КОПЗ_2025

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ 2
ВСТУП..... 3
1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ 5
1.1 Призначення системи..... 5
1.2 Область застосування..... 6
2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ 7
2.1 Огляд існуючих систем, технологій, архітектур та програмних рішень за профілем теми випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти..... 7
2.2 Обґрунтування вибору засобів для побудови системи та мови програмування..... 13
2.3 Розгорнута постановка завдання 15
3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ 16
3.1 Опис функціонування системи 16
3.2 Розробка структурної схеми..... 20
3.3 Розробка функціональної схеми 28
3.4 Розробка діаграми процесів..... 37
4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ..... 39
4.1 Розробка блок-схем та опис алгоритмів функціонування системи..... 39
4.2 Захист розробленого програмного забезпечення..... 52
5 ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ..... 55
6 ОСНОВНІ ВИСНОВКИ..... 61
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 63

КОПЗ_2025

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

БПД	–	бездротова передача даних
ПЗ	–	програмне забезпечення
СПД	–	системи передачі даних
ACK	–	повідомлення підтвердження прийому
ARQ	–	протокол повторної передачі даних
BPSK	–	Binary phase-shift keying
FFD	–	повнофункціональний пристрій
GFSK	–	Gaussian frequency-shift keying
MAC	–	шар механізму доступу
NACK	–	повідомлення непідтвердження прийому
OSI	–	мережна модель
P2P	–	однорангові мережі
PAN	–	персональна мережа
PPS	–	Portable Protocol Stack
RFD	–	пристрій з полегшеними функціями
TDMA	–	часовий поділ
Wi-Fi	–	бездротова технологія

КБПЗ_2025

ВСТУП

Актуальність теми. Попит на мережні камери відеоспостереження й аналітику, що використовує одержувані від них дані, підігрівається усе більше численними погрозами безпеки. Однак очікуваний ріст цього ринку зв'язують насамперед із затребуваністю додатків, призначених для рішення інших завдань.

Згідно із прогнозом Markets and Markets, до 2027 року оберт ринку рішень відеоспостереження досягне 71 млрд доларів, при цьому щорічно він буде збільшуватися на 16,5%. Ключовою тенденцією стає перехід від аналогових рішень до підключення по IP. Попит на мережні камери відеоспостереження й аналітику, що використовує одержувані від них дані, підігрівається зростаючим числом погроз безпеки, зокрема терористичних атак, і жорсткістю законодавчих вимог. І хоча на ці завдання поки доводиться основне число застосувань, очікуваний ріст ринку відеоспостереження й відеоаналітики зв'язують насамперед з додатками, що не ставляться до сфери забезпечення безпеки, такими як вилучений моніторинг у ретейлі, контроль за виробничими процесами, спостереження за пацієнтами й т.п. Виробники пропонують усе більше камер з убудованими аналітичними функціями. На порядок денний виходить необхідність кореляції отриманої інформації з іншими джерелами даних.

Мета й завдання дослідження. Метою роботи є програмне забезпечення системи відеонагляду для IoT.

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

– Огляд існуючих систем відеонагляду для IoT.
– Дослідження системи відеонагляду для IoT.
– Програмна реалізація системи відеонагляду для IoT.

КПЗ-2025

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі відеонагляду для IoT.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи відеонагляду для IoT, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

КБПЗ_2025

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

В 2024 році обсяг продажів мережних IP-відеокамер для організації відеоспостереження вперше перевищив аналогічний показник для традиційних аналогових камер. Незважаючи на появу таких технологій, як аналогове відео високої чіткості (Analog High Definition, AHD), немає ніяких сумнівів в тому, що використання IP-камер для цілей відеоспостереження буде розширюватися завдяки їхнім широким можливостям, які аж ніяк не обмежуються кращим відеорозв'язком. Більше того, їхня корисність може багаторазово зрости за рахунок інтеграції з іншими IP-пристроями забезпечення безпеки в рамках реалізації концепції Інтернету речей. Як затверджують в Axis, компанія випустила перший пристрій для Інтернету речей в 1996 році, коли такого поняття ще не існувало, – це була мережна камера для відеоспостереження. За минулі з тих пор 20 років такі пристрої стали набагато більше ефективними, причому не тільки через поліпшення якості зображення, але й завдяки можливості його обробки безпосередньо на камері – від детектування руху до розпізнавання автомобільних номерів. За рахунок підвищення обчислювальної потужності, відеокамери здатні самостійно обробляти інформацію й приймати рішення, якщо задані сценарії поводження. Як би не було привабливо розширення функціональності окремих пристроїв, їхня спільна робота дозволить вирішувати більше серйозні завдання. У рамках IoT камери можуть обмінюватися даними й приймати рішення без участі посередника (центрального комп'ютера – сервера). Так, наприклад, кілька камер, об'єднаних у мережу, можуть «вісти» підозрілу особу або об'єкт: коли він виходить із сектора огляду однієї, спостереження за ним продовжує наступна. Інший можливий сценарій – узяття під контроль сусіднього сектора, коли яка-небудь камера виявляється тим або іншим способом

КПЗ-2025

осліплена або ушкоджена (для цього пристрій повинне бути оснащене датчиком стороннього втручання). Крім того, камера сама відправить сигнал тривоги охоронцеві, щоб він звернув увагу на сформовану ситуацію.

У міру того як замість колишніх аналогових пристроїв буде використовуватися усе більше пристроїв безпеки з підтримкою IP, IoT дозволить реалізувати комбіновані рішення, що поєднують колись розрізнені пристрої: камери відеоспостереження, детектори диму, пристрою доступу, гучномовці й ін. Axis випустила цілий ряд пристроїв безпеки з підтримкою IP, включаючи гучномовці, мережний відеодомофон і пристрій контролю доступу. Взаємодія пристроїв між собою дозволить скласти більше повну картину того, що відбувається на контрольованому об'єкті.

1.2 Область застосування

Представимо наступний можливий сценарій спільного використання різних пристроїв для охорони периметра. При виникненні позаштатної ситуації – перетинанні охоронюваного периметра потенційним зловмисником у нічний час – тепловізійна камера, що його засікла, відправляє сигнал поворотній камері, після одержання якого та повинна розгорнутися в зазначеному напрямку. Автоматично відправляється оповіщення на мобільний пристрій або робоче місце оператора. Одночасно через гучномовець транслюється звукове попередження про те, що сторонньому необхідно покинути територію. Як свідчать дані alarm.org, у такий спосіб вдається запобігти до трьох чвертей (74%) всіх незавершених вторгнень. У цій схемі немає ніякого комп'ютера – пристрої спілкуються тільки між собою.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи відеонагляду для IoT, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

КПЗ–2025

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

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

Огляд камер відеоспостереження 2024 року

Розв'язна здатність

Навіть поза залежністю від того, що дозвіл уважається найважливішим критерієм при виборі cctv-камери – принаймні, воно є найбільше згадуваємою часто. При цьому відбулися помітні зміни в розумінні даного параметра, які важливі для розуміння суті.

Стандартний дозвіл SD і SD+ (700TVL, 960H, 1000TVL, 1280H та ін.) наказало довго жити

Різні варіації пропозицій на тему SD+, які з'являлися в останні кілька років (разом із широко представленим на ринку розв'язком 960H) – це всього лише недорогі поліпшення систем NTSC / PAL.

До 2018 року всі вони, по суті, зусиллями виробників закінчили своє існування. Причина в тому, що ці камери були витиснуті камерами високого дозволу HD (такими як AHD, CVI, TVI).

Тому намагайтеся уникати камер SD і SD+, якщо тільки це не диктується існуючими вже системами.

Найбільше часто використовуються камери з розв'язком 720p і 1080p

На порозі 2024 року відеокамери з розв'язком 720p і 1080p є найбільш популярними пропозиціями ринку. У цей час дозвіл 720p сприймається як бюджетне рішення, хоча видима різниця в якості між 720p і 1080p є не занадто помітною (незважаючи на подвоєну кількість пікселів). Хоча, з іншого боку, різниця в ціні між камерами 1080p і 720p, як правило, мінімальна.

КОПІЯ 2025

Важливо пам'ятати, що кількість пікселів / дозвіл – це всього лише один з елементів, що впливають на якість зображення. Зокрема, мінімальна освітленість і динамічний діапазон можуть бути істотно різними навіть для відеокамер з однаковим розв’язком.

Новинка – 4 MP

В 2024 році розвитком камер з розв’язком більшим 1080p (2,1MP) будуть нові 4 MP-камери. У цих камер як і раніше "широкоформатне" відношення сторін матриці 16:9, однак у порівнянні з 1080p камерами в них у два рази більше число пікселів. При тестуванні ми виявили не занадто помітну різницю в порівнянні з 1080p, однак багато хто з 4 MP-камер продаються приблизно по тій же ціні, що й камери 1080p. Ці камери заслуговують спеціального розгляду.

Новинка – 4K (8,3 MP)

В 2015 році одержали широку популярність 4K-камери, кількість пікселів у які в 4 рази перевершує подібне значення в 1080p камерах.

Однак у наших тестуваннях функціонування 4K-камер значно варіювалося від моделі до моделі – одні з них були просто жахливими, інші демонстрували значна перевага в порівнянні з камерами 1080p. Тому при виборі 4K-камер варто бути досить уважним, тому що із пропонованих у цей час перших зразків далеко не все є доведеними до розуму. Зокрема, якщо ви берете 4K-камеру без убудованого ІЧ-підсвічування, досить великий ризик одержати камеру з низькою чутливістю.

Багато хто 4K-камери також підтримують 12MP режим, як правило, зі співвідношенням сторін поля зору 4:3 (проти 16:9 для 4K). Використання такого додаткового масштабу може бути корисним, однак це вимагає перевірки. При цьому в сучасних камерах режим 12MP майже завжди пов'язаний з більше низькою частотою кадрів у порівнянні з потоком 4K.

Хоча стандарт 4K припускає "full" (повну) частоту кадрів, однак багато виробників продають свої моделі як 4K-камери, але при значному більше

КОПІЯ 2025

низькому (6...12…12) числі кадрів у секунду. Якщо частота кадрів важлива – перевіряйте цей параметр.

Більше 12 MP

Існує група камер відеоспостереження з розв'язком більше 12 MP. Вибираючи їх, треба бути дуже обережним, оскільки найчастіше в них низьке значення частоти кадрів і низька чутливість, а ціна може бути дуже високої, і при цьому вони вимагають високої пропускної здатності каналу. Ми рекомендуємо подібні камери тестувати прямо на об'єкті, щоб переконатися, що існує істотна різниця при їхньому використанні в порівнянні з камерами 4MP і 4K.

IP vs HD Analog

Торік з'явилися HD-аналогові камери, які можна розглядати як імовірного суперника IP-камер для одержання відео високого дозволу. Можливо, це саме потужне й найбільш важливе рішення за останнім часом.

В HD-аналогових відеокамер є 3 найважливіші переваги:

1. Низька вартість.

2. Простота установки.

3. Часткова сумісність із існуючими рішеннями.

– Ціна HD-аналогових камер звичайно на 30% нижче, ніж в IP-камер з подібними функціями.

– HD-аналогові камери не вимагають ніяких конфігурацій, властивих IP системам, відновлень прошивань або виявлення мережі; вони працюють як працювали звичайні аналогові камери середнього дозволу.

– І нарешті, HD-аналогові відеокамери найчастіше надають можливість повторного використання існуючих на об'єкті коаксіальних кабелів (хоча це обмежується віком кабелів і їхньою довжиною).

Основні недоліки HD-**а**наалогових відеокамер:

1. Обмежений набір функціональних можливостей.

2. Обмеження при виборі постачальника.

3. Обмежена розв'язна здатність.

КПЗ – 2025

4. Несумісності, хоча вони будуть усуватися (принаймні, деякі з них) у найближчі рік або два.

– HD-аналогове відеоспостереження початок свій розвиток з найбільш бюджетного рівня: низька ціна, об'єктиви з фіксованою фокусною відстанню, базові моделі камер. Зараз продавці починають пропонувати широкий динамічний діапазон, надвисоку чутливість, адаптивну ІЧ-підсвічування, автофокус та ін., однак ці особливості, що заявляються продавцями, важливо перевіряти.

– Крім того, багато великих західних і японських компаній просто ігнорують HD-аналогове відеоспостереження. Наприклад, ви не знайдете HD-аналогове встаткування в Axis, Avigilon, Bosch, Panasonic, Pelco, Sony і т.д., і скоріше всього вони не почнуть рухи в цьому напрямку й в 2025 році.

– У цей час розв'язна здатність не перевищує 1080p, хоча TVI обіцяє в 2024 році поява 3MP і 5MP камер, а CVI, як і AHD, уже продемонстрували прототипи майбутніх камер 4K.

– Багато Хто HD-аналогові прилади не сумісні з іншими, до того ж дехто для тих самих технологій використовує різні позначення (приміром, CVI іноді позначається як MPX або HQA і т.д.). Це підвищує ризик того, що різні камери можуть не працювати з реєстраторами.

Якість зображення

Велика кількість пікселів ще не гарантує найкраща якість зображення. Дійсно, при зростанні розв'язної здатності збільшується ризик роботи в широкому динамічному діапазоні й при низької освітленості (наскільки саме – це визначається камерою, але тенденція саме така).

WDR

У минулому році реально широкий динамічний діапазон став застосовуватися значно частіше. Причиною цього з'явилися недорогі сенсори з додаванням даної функції, завдяки чому стала можливої реалізація WDR не тільки в продуктах преміум класу. Однак WDR як і раніше недоступний у

КПІ – 2025

більшості недорогих камер, а виробники нерідко намагаються обдурити, заплутуючи маркетингову термінологію. Тому варто перевіряти, чи реалізується в даній камері WDR при декількох різних експозиціях.

WDR може бути досить корисний у випадках, коли потрібно розглядати деталі зображення (наприклад, особи людини при висвітленні сонячним світлом зі спини – на заході або світанку).

Чутливість

За останні кілька років істотно покращилася технологія обробки зображення при низької освітленості без додавання зовнішніх джерел світла (таких, як ІЧ-підсвічування або вуличне висвітлення). До того ж в 2015 році на ринку з'явилося значне число нових камер з розв'язком 1080p, сенсором 1/2" і сучасними методами обробки зображення. Наше тестування відеокамер показало істотні зрушення в кращу сторону в класі камер з високою чутливістю.

Головним недоліком є лише те, що ці надчуттєві камери з матрицями 1/2" виявляються найдорожчими на ринку.

Убудоване ІЧ-підсвічування

Убудоване ІЧ-підсвічування – це найпоширеніший і недорогий спосіб одержання якісного зображення при низької освітленості, без застосування якого-небудь зовнішнього додаткового підсвічування. За останні кілька років убудована ІЧ-підсвічування перестало бути характерною рисою лише дешевих камер – вона стала пропонуватися практично кожним виробником камер, як дешевих, так і дорогих.

Дистанція ІЧ-підсвічування може становити від 5 м до 50 м і більше. Якість убудованого ІЧ-підсвічування може бути істотно різним. Наприклад, кут підсвічування може не збігатися з кутом огляду камери, що приводить до яркою засветке у вигляді плями в середині кадру. Деякі камери мають адаптовану ІЧ-підсвічування, що виявляє об'єкт і автоматично набудовує потужність ІЧ-підсвічування, щоб виключити надлишкове висвітлення в кадрі.

КПІ – 2025

Відеоаналітика

Більше десяти років у центрі уваги перебуває відеоаналітика камер відеоспостереження.

На жаль, більшість рішень в області відеоаналітики (особливо, поза приміщеннями – саме там, де вона найбільш викликана) працюють не цілком задовільно, викликають велика кількість помилкових тривог. І навіть нові рішення відеоаналітики, такі як, наприклад, недавно запропонована компанією Axis, є усе ще досить слабкими. У принципі можна придбати добре працюючу відеоаналітику, однак навіть в 2024 році варто уважно перевіряти її працездатність у ваших додатках, для ваших цілей (при різній погоді, освітленості, на можливість керування фіктивними тривогами та ін.) – оскільки в масі дана технологія ще не є зрілою.

Зниження цін

Не в останню чергу, звичайно, варто розглядати ціну. На порозі 2024 року можна відзначити, що ціни на камери відеоспостереження рухаються по спадній спіралі, оскільки конкуренти, захоплені "перегонами до дна", знижують ціни. Що є причиною цього – спірно, однак ясно, що ініціюється це китайськими виробниками, починаючи з такого, що просувається урядом мегавиробника, як Hikvision, і закінчуючи дрібними збирачами камер у Шеньчжене. Коли цей процес зупиниться, зараз сказати важко, але траєкторія процесу така, що, імовірно, досить великому числу виробників прийде покинути свій бізнес.

Гарною новиною є те, що цілком пристойні камери можна купити неймовірно дешево (як правило, не дорожче 100 USD). Додайте до того ж, що це камери можуть бути 1/2", 1080p, WDR, 4MP – це краща за увесь час ціна для камер відеоспостереження. Але з іншого боку, ніколи ще не було більше ризиковою справою виробництво й продаж подібних камер.

КПЗ – 2025

2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

Оскільки потрібно розробити просту та легку у користуванні програму, яка б виконувалась під операційною системою Windows, то для її реалізації я обрав Builder C++. Існує велике число бібліотек написаних під Builder C++ , тому це одна з важливих причин вибору мови програмування. Середовище Builder C++ досить просте в користуванні, його вихідний код значно менше по об'єму в порівнянні з Delphi чи деякими іншими програмами такого типу. Досить легко організувати взаємодію між модулями програм, об'єктно-орієнтований підхід дає можливість значно скоротити код програми, а отже і час його виконання.

На заміну старого розробленого набору елементів управління у Builder C++ інтегрована бібліотека візуальних компонентів VCL, представлених на палітрі компонентів. Після переносу на форму методом перетягування (drag-and-drop) компоненти відразу становляться діючими об'єктами вашої програми. Окрім типізованих інтерфейсних елементів Windows (кнопки, смуги прокручування, редагуємі текстові області, прості та комбіновані списки, та інше) у бібліотеку включені елементи підтримки діалогових вікон, обслуговування баз даних та багато іншого. Можливо не тільки модифікувати поведінку існуючих компонентів, але і будувати нові.

Builder C++ підтримує останні розширення стандарту мови C++ та забезпечує швидку компіляцію та складання 32-розрядних програм для Windows. Результуючі програми оптимізовані з точки зору швидкості виконання програм та затрат пам'яті. Зручний відладгоджувальник (з асемблерним вікном, можливістю крокового виконання, завдання точок зупинки, трасування та інше) повністю інтегрований у систему проектування. Дизайнер форм, редактор коду, інспектор об'єктів та інші інструменти зостаються доступними під час виконання програми, саме через це вносити зміни до коду можна прямо у процесі відлагодження.

КОПІЯ 2025

Дизайнер форм, Інспектор об'єктів і інші засоби залишаються доступними під час роботи програми, тому вносити зміни можна в процесі відлагодження.

Builder C++ поставляється в трьох варіантах: Standard (стандартний), Professional (для професіоналів розробників, орієнтованих на мережеву архітектуру) і Client/Server Suite (для розробки систем в архітектурі клієнт/сервер). Останні два варіанти доповнюють стандартний початковими текстами візуальних компонентів, різномасштабним словником даних, новими функціями мови запитів SQL для бази даних, пакетом підтримки систем Internet, службою моніторингу програм, а також рядом інших засобів.

Builder C++ підтримує зв'язок з різними базами даних 3-х видів: dBASE і Paradox: Sybase, Oracle, InterBase і Informix; Excel, Access, FoxPro і Btrieve. Механізм BDE (Borland Database Engine) додає обслуговуванню зв'язків з базами даних дивовижну простоту і прозорість. Провідник Database Explorer дозволяє зображати зв'язки і об'єкти баз даних графічно. Використовуючи компоненти баз даних, я побудував електронний записник згідно таблиці dBASE за півгодини роботи на комп'ютері. Спадкоємство готових форм і їх "підгонка" під специфічні вимоги помітно скорочують тимчасові витрати на вирішення подібних завдань.

Довідкова служба Builder C++ надавала мені допомогу в цій і багатьох інших подібних ситуаціях. Є повний опис кожного управляємого компонента, включаючи списки властивостей і методів, а також численні приклади. Виклад матеріалу в книзі був значно покращуваний і систематизований завдяки відомостям, почерпнутим мною з довідкової служби.

Завдяки засобам управління проектами, двосторонній інтеграції додатку і синхронізації між засобами візуального і текстового редагування, а також вбудованому відладнику (з асемблерним вікном прокрутки, покрокового виконання, точок останову, трасуванням і тому подібне) – Builder C++ корпорації Borland надає собою вражаюче середовище розробки, яка, мабуть, витримає конкурентну боротьбу з такими модними продуктами як Developer Studio фірми Microsoft.

КПЗ-2025

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускну кваліфікаційну роботу за першим (бакалаврським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи відеонагляду для IoT.

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

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

б) вибрати та обґрунтувати методику побудови системи контролю роботи технологічного обладнання на виробництві в автоматизованому режимі. Розробити функціональну та структурну схеми системи;

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

г) організувати інтерфейс користувача з метою формування та виводу на екран ЕОМ повідомлень про некоректні дії користувача та нестандартні ситуації в роботі технологічного обладнання;

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

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

ж) розробити заходи по охороні праці при впровадженні та експлуатації системи, а також розробити заходи з цивільного захисту;

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

КОПЗ-2025

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

Розмаїтість пристроїв в Інтернеті речей уже давно перетворилося в улюблену мету атак хакерів. Їм досить придбати який-небудь девайс і самостійно вивчити, як він захищений. Якщо його зламати, то система захисту не може бути оновлена оперативно. Використовуючи один гаджет, хакеру дуже легко потрапити у всю мережу, що включає банківські реквізити, а також медичні записи, познайомитися з корпоративними даними.

Як ми знаємо, більшу частину гаджетов для Інтернету речей роблять співробітники стартапів, які не можуть захистити себе від зломів, використовують апаратне й програмне забезпечення з боку, придбане в постачальника. На створення більшості IoT-девайсів був витрачено мінімум грошей, тому не дивно, що їх легко зламати.

Із цієї причини розроблювачі стандартів разом з більшими компаніями й стартапами повинні одночасно займатися рішенням найбільш важливих питань безпеки.

Що вже зроблено?

Група стандартів ISO, що працює на міжнародному ринку, займається адаптацією стандартів безпеки ISO 27000 до використання в галузі. А асоціація стандартів IEEE працює над рамковою концепцією, що стандартизує безпеку в сфері IoT і захищає особисті дані.

Крім того створений цілий список галузевих союзів і консорціумів серед розроблювачів IoT.

Наявні проблеми

Хоча прогрес істотний, у сфері захисту IoT існує велика кількість проблем. У звіті міжнародної консалтингової компанії McKinsey & Co. описано

КПЗ-2025

конкретні загальні стандарти безпеки. У той же час інших подібних стандартів поки не існує, або ж функціонує їх трохи, що конкурують між собою.

Програмно-конфігуруємі мережі, що проводять потоки даних IoT, можуть загрожувати безпеки. Цей вид обробки даних складається з великої кількості каналів підключення й застосування вилучених способів обчислення.

У системах беспилотників, над якими сьогодні працюють виробники, іде потік обміну даними між машиною й інфраструктурою доріг, для чого повинна бути створена більше довговічна система безпеки й скорочені шанси проникнути в неї.

Поки головні IT-гіганти не змогли прикласти максимум зусиль, щоб створити рішення для безпеки IoT-додатків. На даний момент цим займаються стартап-компанії, що серйозно розвивають сектор IoT. У консалтинговій компанії Gartner оцінили, що в 2017 році більша частина продуктів IoT буде вироблятися дрібними компаніями, які недавно вийшли на ринок.

Способи захисту майбутнього

Які знання й уміння необхідні новому поколінню розроблювачів, щоб вони могли забезпечити в реальності міцний рівень захисту девайсів IoT?

Наприклад, займатися заохоченням виробників, щоб вони активніше співробітничали з постачальниками ПЗ, апаратного забезпечення й з системой галузі. Старшим партнерам варто допомагати новачкам ринку стати коштовним джерелом для одержання досвіду й знань, які дозволили б застосовувати наявні стандарти й компоненти безпеки.

Необхідно приділити час розвитку утворення, створювати лабораторії безпеки, де розроблювачі будуть мати доступ до важливого встаткування й зроблять свій внесок у розвиток розробок.

В ідеалі ніяких погроз не повинне існувати. Але в цей момент до Інтернету можливе підключення все більша кількість предметів, тому багато хто з них можуть бути потенційно зламані. Поки не ясно, чи може ця проблема бути в принципі вирішена, але об'єднання зусиль посприяє безпеці Інтернету речей.

КОПІЯ 2025

Вважається, що найбільш успішні напрямки бізнесу побудовані на світових тенденціях, які задають темп цілим ринкам. Впливаючи на наше повсякденне життя, ці тенденції стають причиною позитивних змін і розвитку індустрії безпеки.

Відмова від аналогових рішень на користь IP

Наш мир з кожним днем усе більше походить на павутину з IP-адрес, тому «Інтернет речей» обіцяє значні зміни й можливості. Користувачі систем відеоспостереження вже повсюдно відмовляються від аналогових рішень на користь IP технологій, які мають на увазі камери з розширеними можливостями й різноманітне ПЗ для відеоспостереження. Що ще більш важливо, IP-камери дозволяють створити систему з низьковольтними передачею даних і електроживленням.

Ну й нарешті, через різні нещасні випадки й почастішавших злочинів, під безпекою тепер розуміють не тільки спостереження за тим, що відбувається навколо, але й можливість повністю або частково заблокувати доступ до об'єкта. Це значить, що системи відеоспостереження тепер доповнюють системами контролю й керування доступом, і навпаки. Опираючись на ці три тенденції, заповзятливі інтегратори й установники систем безпеки прагнуть запропонувати користувачам комплексні рішення з низьким енергоспоживанням у складі мережних камер і пристроїв для контролю доступу на базі IP технологій.

Корпорація Google неспроста заплатила 3,2 мільярди доларів за компанію-розроблювача «розумних» термостатів і датчиків диму Nest. Очевидно Google у такий спосіб вирішила зайняти нішу на ринку «Інтернету речей», так цей напрямок робить все більший вплив на користувачів систем відеоспостереження й контролю доступу.

По підрахунках дослідницької й консалтингової компанії Gartner, до 2020 року кількість пристроїв «Інтернету речей» досягне 26 мільярдів, що в грошовому еквіваленті може скласти більше 4 трильйонів доларів США. Очікується підвищення попиту на відкриті платформи й багаті користувальницькі

КПЗ-2025

інтерфейси. В 2024 році продажу IP-камер у Північній Америці склали більше 50% від загального обсягу продажів. Різні організації вже випробували популярний в усьому світі тип відеоспостереження, тому що зрозуміли, що завдяки низькому енергоспоживанню можна непогано заощаджувати й по-достоїнству оцінили можливість передачі даних і живлення по одному кабелі Ethernet. Оскільки ключовим поняттям у таких системах є мережа, IT-фахівці стали ефективно співробітничати зі співробітниками служб безпеки.

Важливо, щоб система безпеки, як бездротова, так і з низьковольтними кабелями, був простій і економічно вигідної. Вкладаючи гроші в систему контролю доступу, хочеться замість кімнати з купою панелей керування й кілометрів прокладених проводів одержати креативне рішення. Під креативним рішенням мається на увазі сучасна система з мережними пристроями, що підключаються до Інтернет (і не тільки камери відеоспостереження, але й пристрою для контролю доступу).

Тільки комплексне рішення не залишить злочинцеві шансів

Як уже говорилося, сучасне поняття «безпеки» крім IP-відеоспостереження має на увазі застосування СКД (систем контролю доступу), тобто важливо не тільки побачити, що відбувається у вас на об'єкті, але й мати можливість заблокувати до нього доступ. Подібне комплексне рішення не залишить злочинцеві шансів.

Одним з основних недоліків традиційної аналогової СКД є необхідність прокладати кабелі від дверей до центрального блоку керування. Крім цього, розширення такої системи – задоволення не з дешевих і, найчастіше, викликає ряд складностей. IP-Рішення мінімізує кількість необхідного допоміжного устаткування, для підключення можна використовувати наявну мережу. Зчитувач і контролер об'єднані в одному пристрої, що зберігає в пам'яті всі події. Електроживлення системи забезпечується по кабелі Ethernet, однак на зчитувач/контролер живлення можна подавати традиційним способом через лінійне джерело живлення з акумулятором і незалежно від магнітного або

КПІ – 2025

електромеханічного замка. Це допоможе захистити пристрій від стрибків напруги. Керування системою можна здійснювати за допомогою багатофункціонального спеціалізованого ПЗ, але для повсякденної експлуатації підійдуть і існуючі програми для керування відеоспостереженням, типу Milestone і Video Insight.

Очевидно, що в майбутньому буде спостерігатися подальше збільшення частки IP-Рішень для контролю доступу на ринку. Повний перехід на такі системи займе не так багато часу, як це було з IP-камерами, але наскільки швидко це відбудеться сказати складно. Солідні інтегратори й установники пропонують, як повноцінні IP-Системи для контролю доступу, так і традиційні рішення. Таким чином, можна врахувати індивідуальні особливості кожного об'єкта й підібрати найбільш підходяще рішення.

Світові тенденції спричиняють зміни на ринку, тому подібна стратегія інтеграторів дозволяє їм задовольняти потреби, які диктує ринок. Очікування, покладені на «Інтернет речей», широке застосування IP-камер, а також переосмислення поняття безпеки, під яким тепер мається на увазі інтегрована система для відеоспостереження й керування доступом — все це стимулює попит на подібні рішення й серйозні інтегратори вже сьогодні намагаються це враховувати.

КбПЗ-2025

3.2 Розробка структурної схеми

Як і у випадку хмар, головна проблема для Інтернету речей (і перешкода на шляху його поширення) – сумніву щодо достатньої захищеності рішення, що особливо актуально, коли ця концепція застосовується безпосередньо для рішення завдань забезпечення безпеки. Зламати камеру значно складніше, ніж комп'ютер, – у неї менше інтерфейсів, менше можливостей доступу, більш високий ступінь захисту. Якщо раніше системи безпеки на пристроях були розрізненими, то зараз необхідно створити єдину платформу безпеки (референсну

архітектуру). Для створення такої архітектури свої зусилля об'єднали компанії Axis, Avaya, EMC і Genetec. Стандарти будуть єдиними для всього ланцюжка передачі й обробки відео, починаючи від камер і закінчуючи сховищами.

Що ж очікує нас в області відеоспостереження в контексті Інтернету речей? З ростом обчислювальних можливостей камери будуть ставати усе більше інтелектуальними, а виконувані ними функції, відповідно, усе більше складними й витратними. Багато хто з тих завдань, які зараз зважуються на сервері, стануть виконуватися на камері – у деяких уже реалізованих проектах навіть виконання найпростішого детектування руху дозволив скоротити кількість серверів до чотирьох разів. Усе більше записів буде зберігатися безпосередньо на камері, особливо з появою терабайтних флеш-накопичувачів. Завдяки розширенню функціональних і ресурсних можливостей камер, сценарії їхнього використання стануть набагато більше різноманітними, тим більше що зараз навіть неможливо представити, які ще пристрої з'являться і як вони будуть взаємодіяти між собою.

У рамках IoT камери будуть взаємодіяти між собою й з іншими пристроями при прийнятті рішень без участі посередника. На діаграмі представлена схема взаємодії тепловізора, відеокамери й гучномовця при перетинанні порушником охоронюваного периметра

Відеоаналітика в ретейлі

Поліпшення характеристик камер відеоспостереження, і насамперед якості одержуваного зображення, дозволило використовувати відео, яке знімається, для подальшого аналізу. У результаті спектр завдань, для яких можуть застосовуватися IP-камери, значно розширився й вийшов за межі споконвічної сфери їхнього застосування для забезпечення фізичної безпеки. Як відзначалося в попередньому розділі, усе більше функцій відеоаналітики виконується на камерах, однак, як правило, ця підтримка обмежується найпростішими типовими завданнями. Для виконання ж більше складні й нестандартних потрібні досить значні обчислювальні потужності й ресурси зберігання й, як наслідок, спеціальне серверне програмне забезпечення.

КПЗ-2025

Однієї з найбільш перспективних областей застосування відеоаналітики є роздрібна торгівля. За прогнозом Transparency Market Research, попит на неї буде рости більше швидкими темпами, чим на всі інші види відеоаналітики, у тому числі задіяні в промислових додатках. У ретейлі відеоаналітика може використовуватися для рішення таких завдань, як надання персоналізованого сервісу в рамках програм лояльності, аналіз клієнтури магазина й визначення конверсії (частки відвідувачів, що зробила покупки), дослідження поводження покупців з метою оптимізації розміщення товарів, боротьба зі злодійством і т.д.

Застосування відеоаналітики в бізнесі простіше обґрунтувати – віддача видна відразу. Така ситуація різко контрастує з тої, що характерно для використання відеоспостереження з метою безпеки, через що обґрунтовувати необхідність інвестицій досить складно.

«Сліпі» відеокамери

Поліпшення характеристик і розширення можливостей відеокамер створюють лише необхідні передумови для їхнього ефективного застосування. Коли мова йде про системи безпеки, велике питання, за що платяться гроші». При сприятливому збігу обставин система безпеки на об'єкті ніколи не буде використана й ви навіть не довідаєтеся, гарна у вас система або погана. У противному випадку ви зрозумієте, що система ні на що не годиться, тільки в той момент, коли вона не впорається з поставленими перед нею завданнями.

Остання ситуація скоріше правило, ніж виключення. Проведення експертизи відеозаписів утрудняється тим, що відеокамери в системах відеоспостереження фіксують події на місці події з недостатньою чіткістю, через що не вдається одержати яку-небудь значиму інформацію й установити всі обставини справи.

З яких причин це відбувається? Відеокамери, як правило, мають невисокий дозвіл і події фіксують погано. Іноді якість установленої системи відеоспостереження не контролюється, а користувач не стежить за нею. Відомі випадки, коли встановлювалися не ті засоби відеофіксації: у договорі вказувалися

КОПІЯ 2025

камери з високим розв’язком, а були встановлені пристрої з гіршими параметрами, до того ж не настроєні для виконання необхідних завдань.

Інша проблема полягає в тому, що користувачі прагнуть заощадити на системі відеореєстрації. Багато хто з них, маючи дорогі відеокамери, орієнтуються на максимальну тривалість запису при її низькій якості. У результаті зображення виявляється дуже поганим і зробити ідентифікацію особистості неможливо. Нерідко належна увага не приділяється розташуванню й орієнтації пристроїв, їхньої спрямованості на контрольовані ділянки. Необхідно грамотно розташовувати камери на охоронюваному об'єкті.

Найчастіше для відеозапису й відеореєстрації використовується настільки застаріле обладнання, що зчитувати відеозапис із даних систем уже не вдається. На експертизу часом приносять не відеозапису, а самі відеореєстратори. Проблема в тому, що системи відеореєстрації морально застаріли й строк техпідтримки для них уже минув, так що вважати запису не представляється можливим.

Саме типове завдання – підрахунок відвідувачів. Визначили, скільки людей увійшло в магазин і скільки дійшли до кас, – одержали коефіцієнт конверсії. Далі можна відслідковувати, як цей показник міняється в часі. Рішення досить простої: на стелю вішається камера, і покупці вважаються «по головах». Однак відеоаналітика може використовуватися й для рішення куди більше складних завдань, з якими не вдається ефективно впоратися іншими засобами.

Підприємствам роздрібної торгівлі потрібна інформація про те, що і як людина робить у магазині, скільки часу проводить у того або іншого прилавка, як переміщається між відділами. В ідеалі хотілося б ідентифікувати цього покупця, коли він оплачує покупки. Відеоаналітика на базі розпізнавання осіб таке завдання дозволяє вирішити. Наприклад, фото особи людини можна прив'язати до карти програми лояльності й при пред'явленні карти на касі всі його переміщення по магазині зв'язати в єдину траєкторію, щоб потім проаналізувати. Ці питання звичайно виникають у маркетологів, і на них ми

КПЗ-2025

можемо відповісти за допомогою системи розпізнавання осіб». Додатковий бонус від реалізації систем відеоспостереження й відеоаналітики складається в створенні ефективної системи безпеки. Роздрібним мережам доводиться постійно боротися із дрібними розкраданнями, і застосування системи розпізнавання осіб може бути вкрай вигідно. Вона особливо корисна в магазинах, де спостерігається систематичне злодійство. Злодії діляться на дві категорії: аматори, що відрізняються спонтанним поводженням, і професіонали, які крадуть систематично, причому роблять це в складі стійких груп.

Система розпізнавання осіб дозволяє виявляти такі групи й ефективно з ними боротися. Вона фіксує обличчя всіх вхідних у приміщення. Потім в автоматичному режимі дозволяє встановити, що ті самі люди регулярно перебувають у магазині в те саме час, навіть якщо заходять у нього порізно. Якщо інформація про неї буде занесена в базу даних мережного ретейлера, тільки вони з'являться в будь-якому магазині мережі спрацює сигнал тривоги.

Рисунок 3.1 – Структурна схема системи

Крім роздрібної торгівлі, відеоаналітика широко використовується в промислових додатках. Відповідно до оцінки Transparency Market Research, в

КОПІЯ 2025

2024 році ринок рішень в області машинного зору склав 15,7 млрд доларів, а до 2028 року він повинен вирости майже у два рази – до 28,5 млрд доларів. Технології машинного зору на виробництві використовуються для виявлення дефектів, перевірки якості, виміру компонентів і моніторингу допусків. Збільшенню попиту на такі рішення сприяє повсюдна автоматизація виробництва, оскільки комп'ютеризовані системи здатні виконувати виміри й аналіз більш точно, ніж неавтоматизовані.

Серед стримуючих факторів розвитку цього ринку аналітики відзначають недостачу операторів, що володіють необхідними навичками, і відсутність стандартів в області забезпечення сумісності різних використовуваних пристроїв. Системи відеоспостереження були винайдені для забезпечення безпеки. Яким би не був [охоронюваний] об'єкт, ці завдання однакові, будь те офіс, готель або магазини. Один раз розроблені технології й рішення можуть застосовуватися на мільйонах об'єктів. Виробництво ж все різне, причому зовсім різне – на кожному існує свій виробничий процес. У результаті завдання автоматизації виробничих процесів за допомогою відео-спостереження настільки різні, що для рішення кожної з них необхідна унікальна розробка – стандартними інструментами цього не зробити.

У ретейлі системи відеоспостереження дозволяють ефективно вирішувати більше складні завдання, чим при забезпеченні безпеки: переміщення покупців, підрахунок відвідувачів, оцінка довжини черги. Однак, де б це не робилося, все відбувається однаково, важливо лише дотримати набору умов, щоб одержати результат із заданою точністю. Якщо завдання по забезпеченню безпеки однакові й не залежать від типу об'єктів, як і аналіз для ретейла, то на виробництві виникає безліч специфічних вимог. Як приклади приведемо запити на розробку відеоаналітики для розпізнавання зубця, що відламав, екскаватора (компанія займається розробкою кар'єрів), визначення кількості пляшок у палеті, виявлення великих каменів і сторонніх предметів на стрічці транспортера.

КБПЗ-2025

Безумовно, подібні замовлені розробки будуть коштувати зовсім недешево. Якщо модуль підрахунку відвідувачів потенційно затребуваний на десятках тисяч об'єктів, виходить, його можна продати не дуже дорого. А алгоритм для розпізнавання зубця, що відламав, екскаватора навряд чи знадобиться комусь ще, тому всі інвестиції повинні бути оплачені конкретним замовником. Існує тільки два шляхи виходу із цієї ситуації: не застосовувати технології розвиненого відеоаналізу для виробничих процесів і продовжувати вирішувати ті ж завдання за допомогою низькокваліфікованої робочої сили або платити за розробку дуже більші гроші, що може бути виправдано лише при реалізації великих проектів. (У принципі, можливий і третій шлях – виявити кмітливість, використовувати стандартний модуль нестандартним способом.)

Втім у кожній галузі промисловості можна виділити свій клас типових завдань. Наприклад, при виробництві мікроелектроніки технології машинного зору використовуються для контролю мікросхем і компонентів, таких як опору й конденсатори, в автомобільній галузі – для огляду пофарбованих поверхонь, контролю якості зварювання й перевірки рухового блоку. Відповідно, проблема йому бачиться в іншому: «Економіка в нас такого розміру, що замовник перебуває тільки один і важко що-небудь масштабувати – рішення виходить або поганим, або дуже дорогим». У міру виходу із кризи, сподівається він, з появою типових завдань масового попиту для автоматизації виробництва з'явиться й достатня кількість тих, хто стане їх вирішувати.

Поширення IP-камер створило передумови для розвитку відеоаналітики. Однак завищені очікування відносно можливості рішення всіх завдань відеомоніторингу й здатності аналізувати сцену з тією же ефективністю, що й людина, пішли скоріше на шкоду розвитку цієї технології. У ході проведення круглого стола із проблем відеоаналітики, на початку 2000-х років активно пропагувалася тема розпізнавання осіб – усі розуміли, що ця функція потрібна в системах безпеки, але в силу об'єктивних причин реальні рішення працювали дуже погано. У результаті на якийсь час про цю тему забули, і якщо в середині

КОПЗ-2025

2000-х хтось згадував про розпізнавання осіб, його піднімали на сміх – відношення було таке ж, як до продукції компанії Herbalife. І тільки в останні роки вдалося реанімувати цей напрямок і повернути до нього довіра, оскільки технології дозволили одержувати бажаний результат.

Розвиток відеоаналітики стримується відсутністю однотипних повторюваних завдань. Для якихось прикладних областей удалося виділити таке загальне завдання – розпізнавання осіб. За неї узялися багато команд розроблювачів, і в результаті з'явилося реально працююче рішення. Але, щоб просунутися далі, потрібна ще більша критична маса – ще хоча б одне чітко сформульоване типове завдання, на яку буде запит від мільйонів клієнтів. Під час обговорення з розроблювачами недоліків того або іншого алгоритму часто доводиться чути таке пояснення: якби [у нашого продукту] був мільйон користувачів і вони надсилали б нам претензії щодня по 50 штук, ми б точно довели його до розуму.

Відеоаналітика успішно справляється із простими повторюваними завданнями, для яких передбачені всі можливі сценарії, наприклад з підрахунком відвідувачів. На його думку, головний стримуючий фактор для її широкого поширення – відсутність навченості алгоритмів, оскільки передбачити всі сценарії, наприклад, при виявленні залишених предметів, неможливо. У принципі, для конкретного приміщення протягом дня-двох аналітичну систему можна настроїти таким чином, щоб помилкових спрацьовувань на залишені предмети було небагато. Але проблема в тому, що камер сотні тисяч, якщо не мільйони, і передбачити всі можливі ситуації дуже складно. Якщо аналітика на камері або сервері почне навчатися й розуміти, коли предмет був залишений спеціально, то згодом помилкові спрацьовування вдасться забрати й це буде проривом, але поки таких алгоритмів немає.

Важко пророчити, коли трапиться прорив у тої або іншій технологічній області. Однак навіть без революційних переворотів уже вдається домогтися якісних поліпшень – за рахунок послідовного інкрементального

КПЗ – 2025

вдосконалювання технології, як у випадку розпізнавання осіб, що тепер здійснюється з такою ж точністю, як це робить людина. Відеоаналітика стає придатна для рішення усе більше складних практичних завдань. Відповідні сучасні додатки виконують подвійну функцію: по-перше, вони виявляють критичну інформацію, що ставиться до сфери безпеки, а по-друге, надають дані для бізнес-аналітики й оптимізації ресурсів. Подальший розвиток цього напрямку буде пов'язане з високорівневої інтеграцією аналітичних інструментів з іншими системами, такими як автоматизація будинків, аудіовізуальні системи й IT-Інфраструктура, з метою кореляції інформації з різних джерел для одержання взаємозалежної картини подій.

Потенціал розвитку відеоаналітики величезний і обмежений тільки уявою й креативністю тих, хто використовує ці рішення й розробляє необхідні алгоритми.

3.3 Розробка функціональної схеми

Програмно-апаратний комплекс об’єкту відеонагляду для IoT на базі бездротових мереж є комплектом устаткування, що розміщується на території об’єкту відеонагляду для IoT на базі бездротових мереж та забезпечує первинний збір і обробку відеосигналу, його подальше зберігання та передачу в захищеному вигляді з використанням мережі передавання даних до центральних серверних комплексів.

Конфігурація програмно-апаратного комплексу об’єкту відеонагляду для IoT на базі бездротових мереж включає:

– дві відеокамери з роздільною здатністю не менш як 640 х 480, з можливістю передачі звукового сигналу та швидкістю відеопотоку не менш як 25 кадрів на секунду, можливістю кольорової відеозйомки, настінного/стельового кріплення, а також обов’язковою фіксацією дати та часу зйомки;

КПЗ – 2025

– пристрій для запису та передавання відеосигналу, який здатен забезпечити запис із роздільною здатністю не менш як 640 х 480 із швидкістю відеопотоку не менш як 25 кадрів на секунду, тривалістю не менше 120 годин з кожної камери та передавання відеосигналу до центру обробки даних з роздільною здатністю та швидкістю залежно від якості каналу передавання даних, але не менш як 320 х 240 із швидкістю відеопотоку не менш як 15 кадрів на секунду;

– система безперебійного електропостачання обладнання програмно-апаратного комплексу об’єкту відеонагляду для IoT на базі бездротових мереж протягом не менш як однієї години;

– необхідне комутаційне обладнання та з’єднувальні кабелі (довжина кабелів між камерами та пристроєм для запису і передаванням відеосигналу не повинна перевищувати 15 метрів);

– програмне забезпечення, необхідне для функціонування програмно-апаратного комплексу.

Функціональна схема розробленої системи зображена на рисунку 3.2.

З рисунку видно, що розроблена система складається з наступних частин:

– блок приймання та запису відеосигналу;

– блок обслуговування користувачів та трансляції відеосигналу (веб-сайт).

На етапі розроблення і впровадження системи відеонагляду для IoT функціональні блоки можуть бути об’єднані в довільні логічні функціональні модулі. При такому об’єднанні кожен з модулів повинен бути функціонально незалежним від інших модулів. При цьому допускається створення територіально розподіленої системи функціональних модулів для забезпечення безвідмовності роботи системи відеонагляду для IoT в цілому.

Для підвищення рівня безвідмовності роботи системи відеонагляду для IoT кожен функціональний мережевий вузол центрального серверного комплексу повинен дублюватися. Застосування мережевих технологій і протоколів безвідмовного включення устаткування повинне гарантувати коректну роботу

КБПЗ – 2025

центрального серверного комплексу в разі виходу з ладу одного з двох мережевих пристроїв, з яких складається відповідний функціональний вузол.

Для захисту від мережевих загроз і атак (зокрема типу DoS та DDoS) обладнання центрального серверного комплексу повинне мати відповідне мережеве обладнання. Програмне забезпечення зазначеного обладнання повинне забезпечувати обробку мережевого трафіку на швидкості підключення до мережі передавання даних і мати у своєму складі достатню кількість мережевих інтерфейсів для підключення.

Блок приймання та запису відеоінформації повинен відповідати таким функціональним вимогам:

– безперервне приймання відео- та аудіопотоків з програмно-апаратних комплексів;

– зберігання отриманих даних в архіві відеозаписів протягом не менш як 120 годин з кожної камери відеонагляду для IoT;

– авторизація джерела відеопотоку перед отриманням даних;

– заповнення цілісності відеопотоку шляхом дозавантаження відсутніх фрагментів з архіву програмно-апаратного комплексу;

– наявність можливості перегляду відеозаписів архіву, в тому числі з використанням пошуку.

Блок обслуговування користувачів та трансляції відеосигналу (веб-сайт) повинен відповідати таким функціональним вимогам:

– трансляція відеопотоків для інтернет-користувачів;

– розподіл навантаження за кластером серверів;

– використання кешування для оптимізації завантаження магістральних каналів;

– доступ до веб-порталу з використанням браузерів (Internet 7.0 і вище, Firefox 7.0 і вище, Opera 10.0 і вище, Chrome 10.0.648 і вище) та мобільних пристроїв (Android, iOS);

– попередня реєстрація з можливістю вибору камер відеонагляду для IoT для здійснення перегляду;

КОПЗ_2025

– збереження вибраних камер відеонагляду для IoT у списках відтворення;

– швидке перемикання користувача на одну з попередньо обраних камер відеонагляду для IoT;

– пріоритетний доступ до всіх камер відеонагляду для IoT для групи спеціальних користувачів;

– об’єднання на головній сторінці сайту декількох навігаційних систем;

– пошук за всіма географічними назвами;

– перегляд відеоматеріалів у режимі прямої трансляції з будь-якої попередньо обраної камери відеонагляду для IoT;

– функція додавання камер в обране;

– використання тесту CAPTCHA для забезпечення додаткового рівня захисту від великого потоку запитів;

– використання системи обмеження доступу за набором параметрів (комплексна перевірка, що передбачає обмеження кількості підключень з однієї IP-адреси за визначений період).

Функціональні вимоги до мережі передавання даних

Конфігурація каналів зв’язку мережі передавання даних включає:

– канал зв’язку об’єкту відеонагляду для IoT на базі бездротових мереж для передавання відеосигналу до центрального серверного комплексу з пропускною здатністю за напрямом передавання даних до 512 Кбіт на секунду, інтерфейс підключення на об’єкті відеонагляду для IoT на базі бездротових мереж Ethernet 10/100 Мбіт на секунду (RJ45);

– вхідний канал зв’язку центру обробки даних для отримання відеосигналу з програмно-апаратного комплексу з пропускною здатністю за напрямом отримання даних не менш як 15 Гбіт на секунду;

– вихідні канали зв’язку центру обробки даних для трансляції відеосигналу в Інтернеті із сумарною пропускною здатністю не менш як 80 Гбіт на секунду;

– канал зв’язку між центром обробки даних та центром керування системи відеонагляду для IoT із пропускною здатністю не менш як 15 Гбіт на секунду.

КПІ_2025

Канали зв’язку повинні відповідати таким функціональним вимогам:

– організація тунельних каналів зв’язку між об’єктами відеонагляду для IoT на базі бездротових мереж, центральним серверним комплексом і ситуаційним центром системи відеонагляду для IoT з використанням технологій IP/MPLS, L2VPN, L3VPN та інших;

– забезпечення цілодобового обслуговування передачі даних каналами зв’язку на всіх вузлах мережі передавання даних;

– наявність централізованої диспетчерської служби, що працює в цілодобовому режимі, для обслуговування каналів передачі даних;

– мережа передавання даних у частині центрального серверного комплексу повинна мати повнопов’язану топологію архітектури побудови, яка забезпечує обмін інформацією між об’єктами відеонагляду для IoT на базі бездротових мереж та вузлами центрального серверного комплексу і ситуаційним центром системи відеонагляду для IoT без послуги, що додатково замовляється з високим рівнем надійності;

– зона відповідальності оператора зв’язку – від інтерфейсу мережевого устаткування в центральному серверному комплексі і ситуаційному центрі системи відеонагляду для IoT до інтерфейсу кінцевого устаткування.

Вимоги до ситуаційного центру системи відеонагляду для IoT

Ситуаційний центр системи відеонагляду для IoT повинен забезпечувати виконання завдань з управління процесами функціонування всіх складових частин системи відеонагляду для IoT і нагляд за їх характеристиками, здійснення контролю за відеопотоками, що передаються з веб-камер, параметрів функціонування програмно-апаратних комплексів і складових частин центрального серверного комплексу, а також стану підключення обладнання.

КБПЗ-2025

Блок приймання та запису відеоінформації

- авторизація джерела відеопотоку перед отриманням даних;
- заповнення цілісності відеопотоку шляхом дозавантаження відсутніх фрагментів з архіву програмно-апаратного комплексу;
- безперервне приймання відео- та аудіопотоків з програмно-апаратних комплексів;
- зберігання отриманих даних в архіві відеозаписів протягом не менш як 120 годин з кожної камери відеонагляду для IoT;
- наявність можливості перегляду відеозаписів архіву, в тому числі з використанням пошуку.

Інтерфейс користувача системи відеонагляду для IoT

Блок обслуговування користувачів та трансляції відеосигналу

- швидке перемикання користувача на одну з попередньо обраних камер відеонагляду для IoT;
- пріоритетний доступ до всіх камер відеонагляду для IoT для групи спеціальних користувачів;
- об'єднання на головній сторінці сайту декількох навігаційних систем;
- пошук за всіма географічними назвами;
- перегляд відеоматеріалів у режимі прямої трансляції з будь-якої попередньо обраної камери відеонагляду для IoT;
- функція додавання камер в обране;
- використання тесту CAPTCHA для забезпечення додаткового рівня захисту від великого потоку запитів;
- трансляція відеопотоків для інтернет-користувачів;
- розподіл навантаження за кластером серверів;
- використання кешування для оптимізації завантаження магістральних каналів;
- доступ до веб-порталу з використанням браузерів та мобільних пристроїв (Android, iOS);
- попередня реєстрація з можливістю вибору камер відеонагляду для IoT;
- збереження вибраних камер відеонагляду для IoT у списках відтворення;
- використання системи обмеження доступу за набором параметрів (комплексна перевірка, що передбачає обмеження кількості підключень з однієї IP-адреси за визначений період).

Рисунок 3.2 – Функціональна схема системи

КОПІЯ 2025

У разі виявлення некоректного функціонування системи відеонагляду для IoT ситуаційний центр повинен визначити джерело несправності.

Функціональними вимогами до ситуаційного центру системи відеонагляду для IoT є:

– приймання діагностичної інформації про складові частини системи відеонагляду для IoT;

– відображення діагностичної інформації на всіх пристроях системи відеонагляду для IoT;

– здійснення нагляду за змінами заданого набору параметрів;

– забезпечення зворотного зв'язку із службою технічної підтримки для розв'язання проблем.

Моніторинг центрального серверного комплексу

Моніторинг центрального серверного комплексу здійснюється шляхом внутрішнього контролю за працездатністю серверного обладнання.

Серверний комплекс повинен мати такі параметри:

– стан використання пам'яті;

– доступність у мережі;

– кількість зайнятого місця за розділами жорстких дисків;

– кількість підключених каналів (камер).

Моніторинг програмно-апаратного комплексу

Під час проведення моніторингу програмно-апаратного комплексу встановлюється наявність таких відомостей:

1. У період здійснення відеотрансляцій для кожної об'єкту відеонагляду для IoT на базі бездротових мереж:

– номер об'єкту відеонагляду для IoT на базі бездротових мереж;

– місце її розташування;

– мережева адреса об'єкту відеонагляду для IoT на базі бездротових мереж;

– номінальна швидкість доступу до мережі передачі даних;

КОПЗ_2025

– фактична швидкість доступу до мережі передачі даних;

– наявність доступу програмно-апаратного комплексу до мережі передачі даних;

– тривалість періоду доступу програмно-апаратного комплексу до мережі передачі даних або його відсутності;

– час початку зйомки і збирання записів;

– час закінчення зйомки і збирання записів;

– сумарна тривалість зібраних записів (час);

– сумарний об’єм зібраних записів (Мбайт).

2. У період здійснення відеотрансляцій для повідомлень про проблеми, що виникають під час перегляду відеотрансляцій:

– кількість повідомлень про проблеми перегляду відеотрансляцій;

– кількість відеокамер, стосовно яких надійшли повідомлення про проблеми, що виникають під час перегляду відеотрансляцій;

– кількість відеокамер, стосовно яких повідомлення про проблеми перегляду відеотрансляцій були підтверджені.

3. У період здійснення відеотрансляцій для єдиного порталу:

– доступність/недоступність єдиного порталу в Інтернеті (період проведення перевірки – не більше 10 хвилин);

– тривалість періоду доступності єдиного порталу в Інтернеті або її відсутності.

4. У період зберігання зібраних записів відеотрансляцій:

– сумарна тривалість записів відеотрансляцій, зібраних у процесі роботи системи відеонагляду для IoT (час);

– сумарний об’єм записів відеотрансляцій, зібраних у процесі роботи системи відеонагляду для IoT (Мбайт);

– сумарна тривалість записів відеотрансляцій, що зберігаються на відповідний час;

– сумарний об’єм записів відеотрансляцій на даний час (Мбайт).

КОПЗ-2025

Вимоги до забезпечення цілісності інформаційних ресурсів та захисту інформації від несанкціонованого доступу

Для унеможливлення несанкціонованого доступу до системи відеонагляду для IoT, використання інформації не за призначенням та порушення її цілісності створюється система захисту інформації від несанкціонованого доступу.

Захист інформації в системі відеонагляду для IoT забезпечується відповідно до Правил забезпечення захисту інформації в інформаційних, телекомунікаційних та інформаційно-телекомунікаційних системах, затверджених постановою Кабінету Міністрів України від 29 березня 2006 р. № 373.

Захист інформації здійснюється за такими принципами:

– безперервність процесу захисту шляхом забезпечення захисту інформації від несанкціонованого доступу протягом усього періоду функціонування системи відеонагляду для IoT;

– керованість процесу захисту шляхом забезпечення оперативних режимів контролю за станом виконання правил доступу до системи відеонагляду для IoT та прийняття відповідних рішень у разі порушення таких правил;

– комплексне використання засобів та методів захисту інформації від несанкціонованого доступу шляхом спільного використання технічних, програмних, організаційних засобів захисту, а також відповідних правил доступу до інформації.

Система захисту інформації від несанкціонованого доступу повинна виконувати такі функції:

– ідентифікація користувачів мережі, захист параметрів ідентифікації користувачів і параметрів;

– розмежування доступу до ресурсів і процесів у системі відеонагляду для IoT;

– забезпечення цілісності інформаційних ресурсів;

– захист носіїв інформації від несанкціонованого доступу;

КОПІЯ 2025

– забезпечення функцій адміністрування в системі захисту інформації від несанкціонованого доступу;

– захист від шкідливих комп'ютерних програм.

Необхідно визначити комплекс адміністративних заходів для дотримання відповідних вимог до рівня надійності приміщень, в яких розташовуються елементи системи відеонагляду для IoT, та регламентації дій персоналу.

Розглянувши усі блоки функціональної схеми перейдемо до розгляду діаграми взаємодії процесів, які відбуваються у системі.

3.4 Розробка діаграми процесів

Діаграма процесів розробленої системи зображена на рисунку 3.3. При детальному її розгляді можна побачити як саме проходе взаємодія у розробленій системі.

КПЗ - 2025

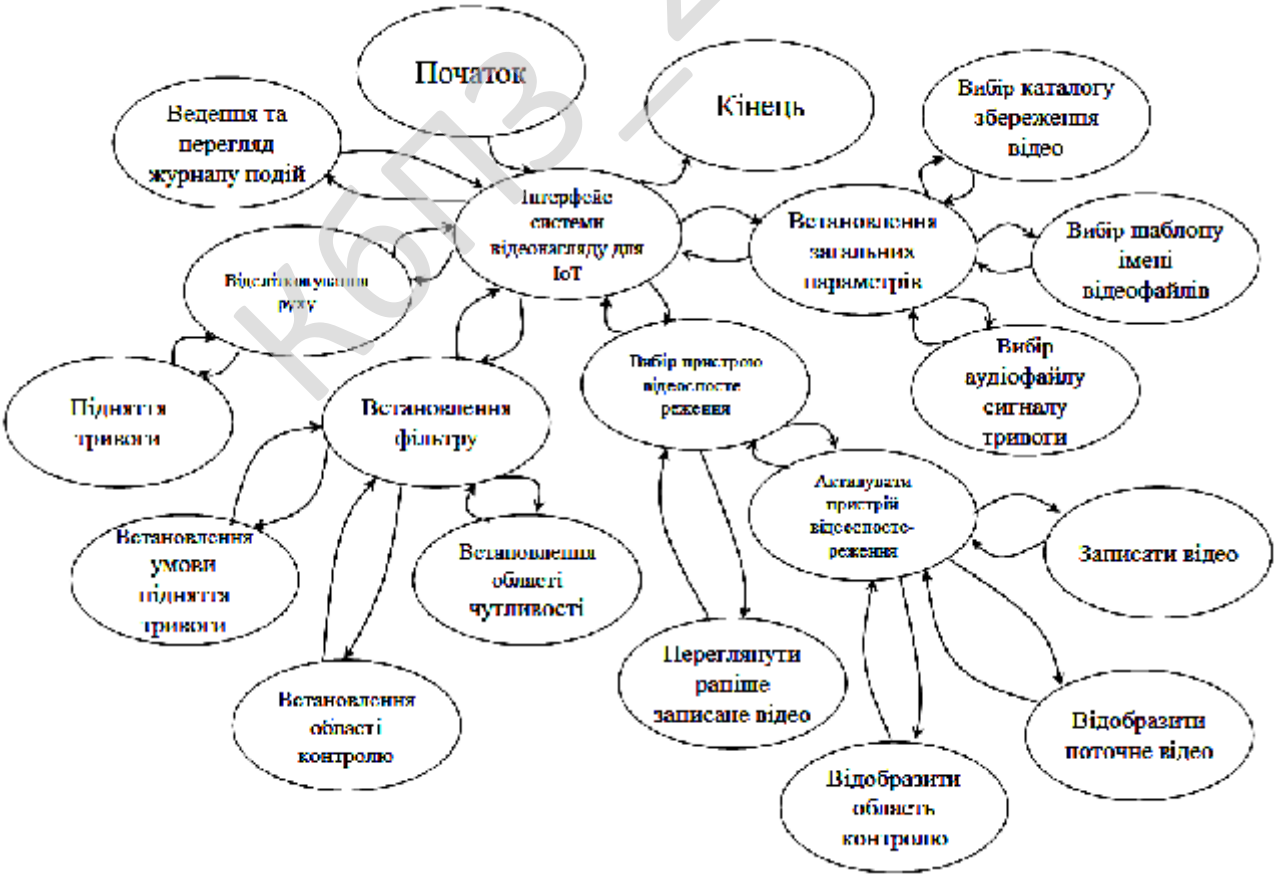


Рисунок 3.3 – Діаграма взаємодії процесів

Використовується модель проектування, графічне представлення «потоків» даних в інформаційній системі. Діаграма взаємодії процесів використовується для візуалізації процесів обробки даних (структурне проектування). Для розробника вважається звичним спочатку креслити діаграму взаємодії процесів даних рівня контексту, завдяки чому буде показано взаємодію системи.

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

Діаграми потоків даних містять чотири типи елементів:

– Процеси які являють собою трансформацію даних в рамках описуваної системи.

– Сховища даних (репозиторії).

– Зовнішні по відношенню до системи сутності.

– Потоки даних між елементами трьох попередніх типів.

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

КбПЗ-2025

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

4.1 Блок-схеми та опис алгоритмів функціонування системи

Блок-схеми є основою ПЗ. Тому від точності і детальності проробки блок-схеми залежить результат всієї програми.

При виборі початкової точки відліку при побудові схем було враховано, що виходячи з вибору мови програмування і інших технічних засобів, програма буде об’єктно-орієнтована що вимагає оптимізації, також те, що при розробці програми слід надати особливу увагу модулю системи відеонагляду для IoT.

Функціональні блоки на схемі позначають прямокутниками, всередині яких надписують їх найменування відповідно до функцій, що виконуються. Зв'язки між функціональними блоками (внутрішні впливи) позначаються лініями зі стрілками, які вказують напрям впливів.

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

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

З яких видно що робота основної програми складається з початкових етапів ініціалізації ПЗ, перевірки наявності ресурсів системи, блоку початку основного циклу з чеканням запиту від користувача в якому відбувається виклик підпрограми та останньої стадії – перевірки поточного стану та поверненням на початок схеми чи з завершенням роботи розробленого ПЗ.

КбПЗ_2025

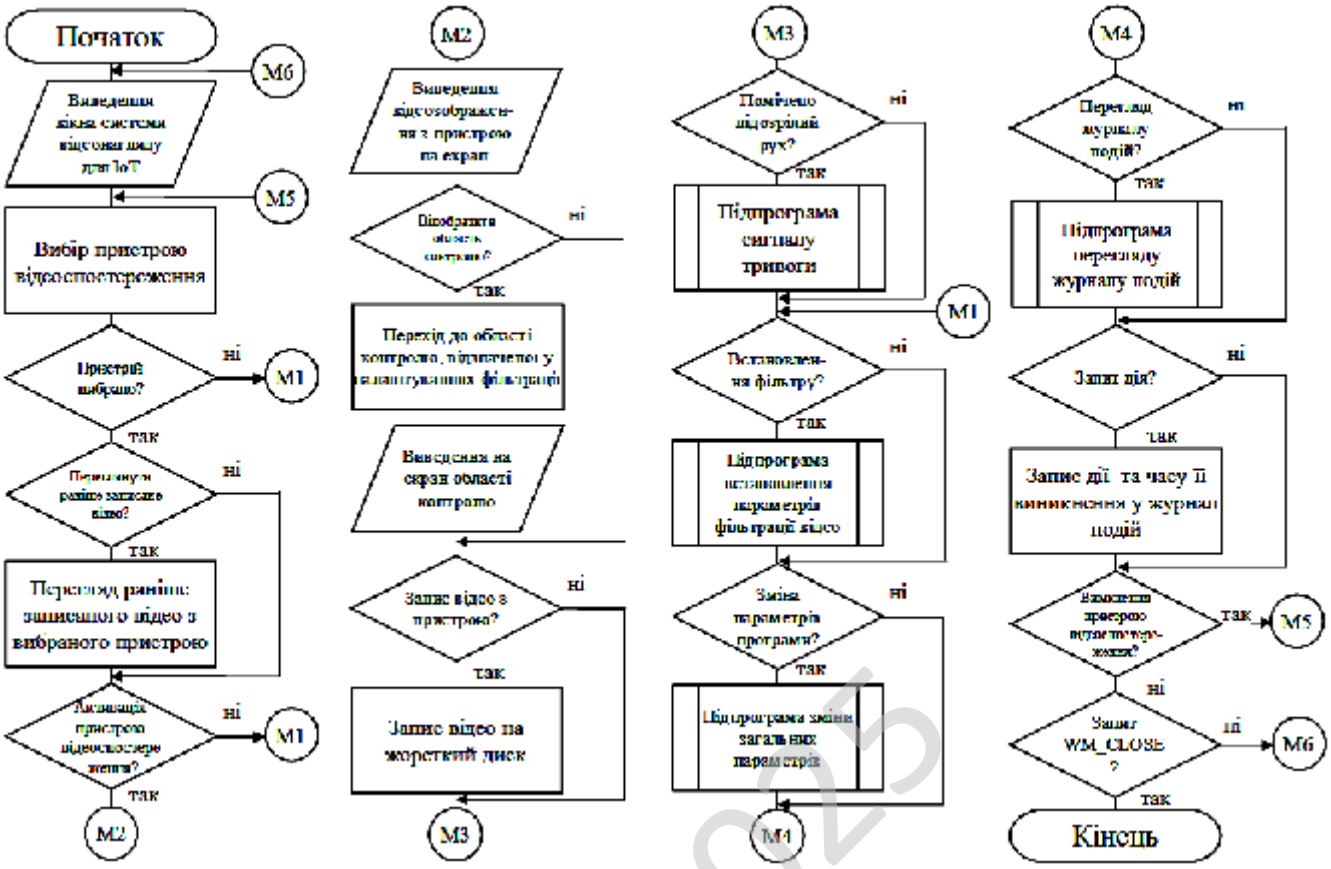


Рисунок 4.1 – Блок-схема основної програми

При роботі підпрограми виконується основний функціонал системи з циклічними послідовностями, перевіркою поточного стану та поверненням в основну програму прапорів стану виконання.

Було використано підходи з використанням UML, це уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого програмування. Є невід'ємною частиною уніфікованого процесу розробки програмного забезпечення. UML є мовою широкого профілю, це відкритий стандарт, що використовує графічні позначення для створення абстрактної моделі системи, називаної UML-моделлю. UML був створений для визначення, візуалізації, проектування й документування в основному програмних систем. UML не є мовою програмування, але в засобах виконання UML-моделей як інтерпретованого коду можлива кодогенерація.

КПЗ_2025

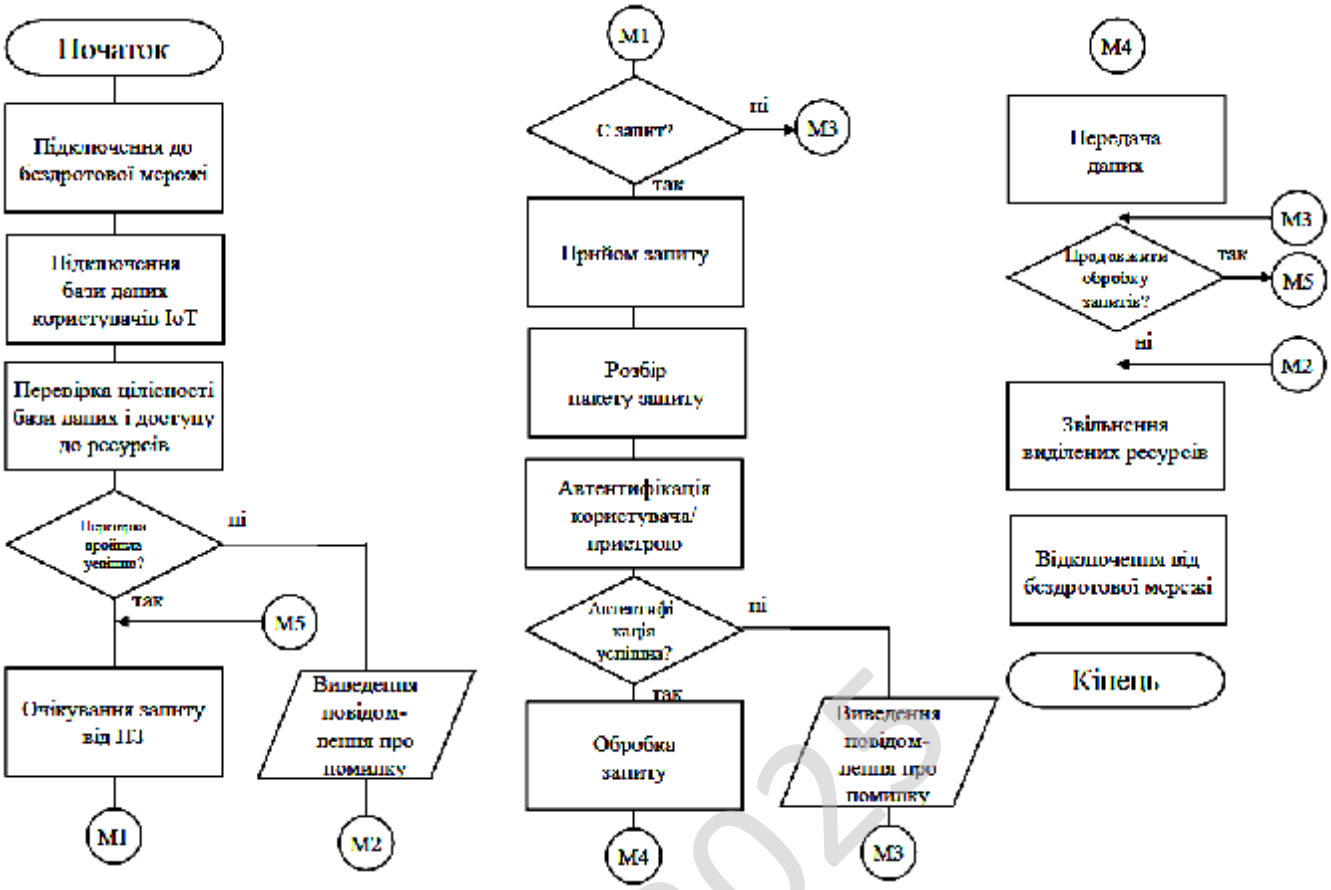


Рисунок 4.2 – Блок-схема роботи підпрограми

Розглянемо використані технології та їх основні компоненти що підтверджують правильність використаних проектних рішень. Наведемо частину коду підпрограми вибору пристрою для відеоспостереження:

```
TSetDeviceForm *SetDeviceForm;
//=====
__fastcall TSetDeviceForm::TSetDeviceForm(TComponent* Owner)
        : TForm(Owner)
{}
//=====
void __fastcall TSetDeviceForm::FormShow(TObject *Sender)
{
  //зберегти поточні налаштування
  Sets.SaveToFile("prev.cfg");

  UpdateState();
  // перерахування пристроїв
```

КПЗ_2025

```
  char szDeviceName[80];
  char szDeviceVersion[80];
  DeviceSelect->Items->Clear();
  for (int wIndex = 0; wIndex < 10; wIndex++)
  {
    if (capGetDriverDescription (wIndex, szDeviceName,
        sizeof (szDeviceName), szDeviceVersion,
        sizeof (szDeviceVersion)))
    {
        // Додати ім’я до списку встановлених драйверів захвату
        // та потім дозволити користувачеві вибрати драйвер для використання
        DeviceSelect->Items-
>Add(AnsiString(szDeviceName)+"("+AnsiString(szDeviceVersion)+")");
        if(Sets.DevIndex == wIndex) DeviceSelect->Text =
AnsiString(szDeviceName)+"("+AnsiString(szDeviceVersion)+")";
    }
  }
}
//=====
void __fastcall TSetDeviceForm::DeviceSelectSelect(TObject *Sender)
{
   //update device index
   Sets.DevIndex = DeviceSelect->ItemIndex;
   UpdateState();
}
//=====
void __fastcall TSetDeviceForm::OkBitBtnClick(TObject *Sender)
{
  DeleteFile("prev.cfg");
  Close();
}
//=====
void __fastcall TSetDeviceForm::CancelBitBtnClick(TObject *Sender)
{
  Sets.LoadFromFile("prev.cfg");
  DeleteFile("prev.cfg");
  Close();
}
//=====
void __fastcall TSetDeviceForm::VideoSrcBitBtnClick(TObject *Sender)
{
   capDlgVideoSource(Sets.hCaptureW);
```

КбПЗ_2025

```
}
//=====
void __fastcall TSetDeviceForm::FormatBitBtnClick(TObject *Sender)
{
   capDlgVideoFormat(Sets.hCaptureW);
   capGetStatus(Sets.hCaptureW, &Sets.CapStatus, sizeof (CAPSTATUS));

   Sets.VideoFormatSize = capGetVideoFormatSize(Sets.hCaptureW);
   if(Sets.VideoFormat){
      delete[] Sets.VideoFormat;
      Sets.VideoFormat = NULL;
   }
   if(Sets.VideoFormatSize > 0){
      Sets.VideoFormat = (LPBITMAPINFO) new BYTE[Sets.VideoFormatSize];
      capGetVideoFormat(Sets.hCaptureW, Sets.VideoFormat, Sets.VideoFormatSize);
   }
}
//=====
void __fastcall TSetDeviceForm::DisplayBitBtnClick(TObject *Sender)
{
   capDlgVideoDisplay(Sets.hCaptureW);
}
//=====
void TSetDeviceForm::UpdateState()
{
   //під'єднати новий пристрій
   capDriverDisconnect (Sets.hCaptureW);
   bool fOK = capDriverConnect(Sets.hCaptureW, Sets.DevIndex);
   if(fOK){
      //отримати заголовки драйверу
      capDriverGetCaps(Sets.hCaptureW, &Sets.CapDrvCaps, sizeof (CAPDRIVERCAPS));
      //отримати заголовки вікна
      capGetStatus(Sets.hCaptureW, &Sets.CapStatus, sizeof (CAPSTATUS));
      //отримати доступні діалоги для драйверу
      // діалогове вікно для джерела відеосигналу
      VideoSrcBitBtn->Visible = false;
      if (Sets.CapDrvCaps.fHasDlgVideoSource)
         VideoSrcBitBtn->Visible = true;
      // діалогове вікно для формату відео
      FormatBitBtn->Visible = false;
      if (Sets.CapDrvCaps.fHasDlgVideoFormat)
         FormatBitBtn->Visible = true;
```

КОПЗ_2025

```
      // діалогове вікно для відеозображення
      DisplayBitBtn->Visible = false;
      if (Sets.CapDrvCaps.fHasDlgVideoDisplay)
         DisplayBitBtn->Visible = true;
   }
}
```

Наведемо частину коду підпрограми для фільтрації відеозображення:

```
TSetFilterForm *SetFilterForm;
int SRX1, SRX2, SRY1, SRY2;
int CurSens = -1;
//=====
__fastcall TSetFilterForm::TSetFilterForm(TComponent* Owner): TForm(Owner){}
//=====
void __fastcall TSetFilterForm::OkBitBtnClick(TObject *Sender)
{
   DeleteFile("prev.cfg");
   Close();
}
//=====
void __fastcall TSetFilterForm::CancelBitBtnClick(TObject *Sender)
{
   Sets.LoadFromFile("prev.cfg");
   DeleteFile("prev.cfg");
   Close();
}
//=====
void __fastcall TSetFilterForm::SRPBPaint(TObject *Sender)
{
   DrawSR();
}
//=====
void __fastcall TSetFilterForm::SRPBMouseMove(TObject *Sender,
      TShiftState Shift, int X, int Y)
{
  if(DrawRectButton->Down){
     SRPB->Cursor = crCross;
     if(Shift.Contains(ssLeft)){//натиснута ліва клавіша миші
        SRX2 = X;
        SRY2 = Y;
        DrawSR();
     }
```

КБПЗ_2025

```
  }
  else{
     SRPB->Cursor = crDefault;
  }
}
//=====
void __fastcall TSetFilterForm::SRPBMouseDown(TObject *Sender,
      TMouseButton Button, TShiftState Shift, int X, int Y)
{
   if(Button == mbLeft && DrawRectButton->Down)
   {
      SRX1 = SRX2 = X;
      SRY1 = SRY2 = Y;
   }
}
//=====
void TSetFilterForm::DrawSR()
{
    TPicture *tPic = new TPicture;
    tPic->Bitmap->Width = SRPB->Width;
    tPic->Bitmap->Height =  SRPB->Height;
    tPic->Bitmap->Canvas->Brush->Color = (TColor)RGB(100,100,100);
    tPic->Bitmap->Canvas->FillRect(tPic->Bitmap->Canvas->ClipRect);
    tPic->Bitmap->Canvas->Brush->Color = clWhite;
    tPic->Bitmap->Canvas->FillRect(Rect(SRX1, SRY1, SRX2, SRY2));
    tPic->Bitmap->Canvas->CopyMode = cmSrcAnd;
    tPic->Bitmap->Canvas->StretchDraw(tPic->Bitmap->Canvas->ClipRect,
Sets.FramePic->Bitmap);
    SRPB->Canvas->Draw(0, 0, tPic->Bitmap);
    delete tPic;
}
void __fastcall TSetFilterForm::FormShow(TObject *Sender)
{
   //зберегти поточні налаштування
   Sets.SaveToFile("prev.cfg");
   SRPB->Width = Sets.pW;
   SRPB->Height = Sets.pH;
   SensPB->Width = Sets.pW;
   SensPB->Height = Sets.pH;
   if(Sets.ClipRect.bottom > 0){
      SRX1 = Sets.ClipRect.left;
      SRX2 = Sets.ClipRect.right;
```

КОПІЯ 2025

```
      SRY1 = Sets.ClipRect.top;
      SRY2 = Sets.ClipRect.bottom;
   }else{
      SRX1 = SRY1 = 0;
      SRX2 = Sets.pW;
      SRY2 = Sets.pH;
   }
   ActivePointsEdit->Text = IntToStr(Sets.MaxActivePoints);
   AlarmTimeEdit->Text = IntToStr(Sets.AlarmMotionTime);
   CycleTimeEdit->Text = IntToStr(Sets.CycleTime);
   ForceSaveEdit->Text = IntToStr(Sets.ForceSaveTime);
}
//=====
void __fastcall TSetFilterForm::RectSaveButtonClick(TObject *Sender)
{
   int t;
   if(SRX1 > SRX2){
      t = SRX1; SRX1 = SRX2; SRX2 = t;
   }
   if(SRY1 > SRY2){
      t = SRY1; SRY1 = SRY2; SRY2 = t;
   }
   Sets.ClipRect.left = SRX1;
   Sets.ClipRect.right = SRX2;
   Sets.ClipRect.top = SRY1;
   Sets.ClipRect.bottom = SRY2;
   if(Sets.bConnected) capGrabFrame(Sets.hCaptureW);
}
//=====
void __fastcall TSetFilterForm::SensPBPaint(TObject *Sender)
{
   DrawSensRegions();
}
//=====
void TSetFilterForm::DrawSensRegions()
{
   int i;
   SensRegion *sR;
   TPicture *tPic = new TPicture;
   tPic->Bitmap->Width = SensPB->Width;
   tPic->Bitmap->Height =  SensPB->Height;
   for(i = 0; i < Sets.SensList->Count; i++){
```

КбПЗ_2025

```
      sR = (SensRegion*)Sets.SensList->Items[i];
      int cl = sR->sens * 255.0 / 100.0;
      tPic->Bitmap->Canvas->Brush->Color = (TColor)RGB(cl, cl, cl);
      tPic->Bitmap->Canvas->FillRect(*sR);
   }
   tPic->Bitmap->Canvas->CopyMode = cmSrcAnd;
   tPic->Bitmap->Canvas->Draw(0, 0, Sets.GrayPic->Bitmap);
   tPic->Bitmap->Canvas->Font->Color = clRed;
   tPic->Bitmap->Canvas->Pen->Width = 1;
   for(i = 0; i < Sets.SensList->Count; i++){
      sR = (SensRegion*)Sets.SensList->Items[i];
      tPic->Bitmap->Canvas->Brush->Style = bsSolid;
      tPic->Bitmap->Canvas->Brush->Color = clRed;
      tPic->Bitmap->Canvas->FrameRect(*sR);
      tPic->Bitmap->Canvas->Brush->Style = bsClear;
      tPic->Bitmap->Canvas->TextOut((sR->left+sR->right)*0.5-10, (sR->top+sR-
>bottom)*0.5-10, IntToStr(sR->sens));
   }
   SensPB->Canvas->Draw(0, 0, tPic->Bitmap);
   delete tPic;
  // SensPB->Canvas->Draw(0, 0, Sets.GrayPic->Bitmap);
}
void __fastcall TSetFilterForm::SensSpeedButtonClick(TObject *Sender)
{
   if(!((TSpeedButton*)Sender)->Down){
      CurSens = -1;
   }else{
      CurSens = StrToInt(((TSpeedButton*)Sender)->Caption);
   }
}
//======
void __fastcall TSetFilterForm::SensPBMouseUp(TObject *Sender,
      TMouseButton Button, TShiftState Shift, int X, int Y)
{
   int i;
   SensRegion *sR;
   if(CurSens != -1 && Button == mbLeft){
      for(i = 0; i < Sets.SensList->Count; i++){
         sR = (SensRegion*)Sets.SensList->Items[i];
         if(sR->left < X && sR->right > X && sR->top < Y && sR->bottom > Y){
            sR->sens = CurSens;
         }
```

КОПЗ_2025

```
      }
      DrawSensRegions();
   }
}
//=====
void __fastcall TSetFilterForm::SensPBMouseMove(TObject *Sender,
      TShiftState Shift, int X, int Y)
{
   if(CurSens != -1){
      SensPB->Cursor = crHandPoint;
   }else{
      SensPB->Cursor = crDefault;
   }
}
//=====
void __fastcall TSetFilterForm::SensClearButtonClick(TObject *Sender)
{
   int i;
   SensRegion *sR;
   for(i = 0; i < Sets.SensList->Count; i++){
      sR = (SensRegion*)Sets.SensList->Items[i];
      sR->sens = 90;
   }
   DrawSensRegions();
}
//=====
void __fastcall TSetFilterForm::AlarmApplyBitBtnClick(TObject *Sender)
{
   try{
      Sets.MaxActivePoints = StrToInt(ActivePointsEdit->Text.Trim());
      Sets.AlarmMotionTime = StrToInt(AlarmTimeEdit->Text.Trim());
      Sets.CycleTime = StrToInt(CycleTimeEdit->Text.Trim());
      Sets.ForceSaveTime = StrToInt(ForceSaveEdit->Text.Trim());
   }catch(...){
      ShowMessage("Введене число не входить в допустимий діапазон значень!");
   }
}
```

Архітектура клієнт-сервер є одним із архітектурних шаблонів програмного забезпечення та є домінуючою концепцією у створенні розподілених

КбПЗ_2025

мережних програм і передбачає взаємодію та обмін даними між ними. Вона передбачає такі основні компоненти:

– набір серверів, які надають інформацію або інші послуги програмам, які звертаються до них;

– набір клієнтів, які використовують сервіси, що надаються серверами;

– мережа, яка забезпечує взаємодію між клієнтами та серверами.

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

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

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

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

– рівень представлення даних, який по суті являє собою інтерфейс користувача і відповідає за представлення даних користувачеві і введення від нього керуючих команд;

КПЗ – 2025

– прикладний рівень, який реалізує основну логіку ПЗ і на якому здійснюється необхідна обробка інформації;

– рівень управління даними, який забезпечує зберігання даних та доступ до них.

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

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

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

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

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

Але, крім браузерів, до серверів можуть звертатися і інші клієнти, а саме – автономні програми. Вони можуть передбачати взаємодію з людиною, а можуть працювати в цілком автоматичному режимі. Типовим класом таких програм є роботи, призначені для автоматичного перегляду веб-ресурсів. Зокрема, роботи є

КОПЗ_2025

важливим елементом пошукових систем і використовуються ними для перегляду сторінок і збору інформації про них.

Для запиту до веб-сервера клієнтська програма повинна задати місцезнаходження комп'ютера, на якому розміщується серверна програма, назву потрібного документа і, можливо, інші дані, які специфікують запит. Мережа забезпечує знаходження сервера і передачу йому клієнтського запиту. Серверні програми обробляють цей запит, відповідь пересилається по мережі клієнтові.

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

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

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

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

КОПЗ-2025

обробки. Дані, з якими працює сервер даних, найчастіше організовані як реляційна база даних.

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

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

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

4.2 Захист розробленого програмного забезпечення

Для захисту розробленого програмного забезпечення запропоновано використовувати алгоритм MMB, в основі якого лежить змішування операцій різних алгебраїчних груп. MMB – ітеративний алгоритм, що складається з лінійних дій (XOR і використання ключа) і паралельного застосування чотирьох великих оборотних нелінійних підстановок. Ці підстановки визначаються за допомогою множення по модулю 2^{32}-1 з постійними множниками. У підсумку з'являється алгоритм, що використовує 128-бітовий ключ і 128-бітовий блок.

КПЗ-2025

Алгоритм MMB оперує 32-бітовими підблоками тексту (x_0, x_1, x_2, x_3) і 32-бітовими підблоками ключу (k_0, k_1, k_2, k_3). Це спрощує реалізацію алгоритму на сучасних 64-бітових процесорах. Чергуючись із операцією XOR, шість разів використовується нелінійна функція *f*. Запишемо операції алгоритму (всі операції з індексами виконуються по модулю 4):

$$x_i = x_i \oplus k_i \text{ для } i = 0..3$$

$$f(x_0, x_1, x_2, x_3)$$

$$x_i = x_i \oplus k_{i+1} \text{ для } i = 0..3$$

$$f(x_0, x_1, x_2, x_3)$$

$$x_i = x_i \oplus k_{i+2} \text{ для } i = 0..3$$

$$f(x_0, x_1, x_2, x_3)$$

$$x_i = x_i \oplus k_i \text{ для } i = 0..3$$

$$f(x_0, x_1, x_2, x_3)$$

$$x_i = x_i \oplus k_{i+1} \text{ для } i = 0..3$$

$$f(x_0, x_1, x_2, x_3)$$

$$x_i = x_i \oplus k_{i+2} \text{ для } i = 0..3$$

$$f(x_0, x_1, x_2, x_3)$$

Функція *f* виконується в три кроки:

1. $x_i = c_i * x_i$ для $i = 0..3$ (Якщо на вході множення одні одиниці, то на виході – теж одні одиниці).

2. Якщо молодший значущий біт $x_0 = 1$, то $x_0 = x_0 \oplus C$. Якщо молодший значущий байт $x_3 = 0$, то $x_3 = x_3 \oplus C$.

3. $x_i = x_{i-1} \oplus x_i \oplus x_{i+1}$ для $i = 0..3$.

Всі операції з індексами виконуються по модулю 4. Операція множення на кроці 1 виконується по модулі 2^{32}-1. Спеціальний випадок для даного алгоритму: якщо другий операнд дорівнює 2^{32}-1, результат теж дорівнює 2^{32}-1. В алгоритмі використовуються наступні константи:

C = 2aaaaaaa, c_0 = 025f1cdb, $c_1 = 2*c_0$, $c_2=2^3 *c_0$, $c_3=2^7 *c_0$.

КОПЗ_2025

Константа C – «найпростіша» константа без кругової симетрії, високою трійковою вагою й нульовим молодшим значущим бітом. У константи c_0 є інші особливі характеристики. Константи c_1, c_2 і c_3 – зрушені версії c_0, і служать для запобігання атак, заснованих на симетрії.

Розшифрування виконується у зворотному порядку, Етапи 2 і 3 інверсні їм самим. На етапі 1 замість c_i використовується c_i^{-1}. Значення c_0^{-1} = 0dad4694.

КБПЗ_2025

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

Розглянемо розроблене ПЗ системи відеонагляду для IoT яке зображено на рисунку 5.1а та вікна налаштувань 5.1б. З рисунку можна побачити що інтерфейс головного вікна розподілено на наступні функціональні розділи:

– Навігаційне меню: Дії; Вид; Довідка.

– Вікно виведення результату роботи системи.

– Навігаційного меню яке викликається натисканням правої клавіші маніпулятора миші.

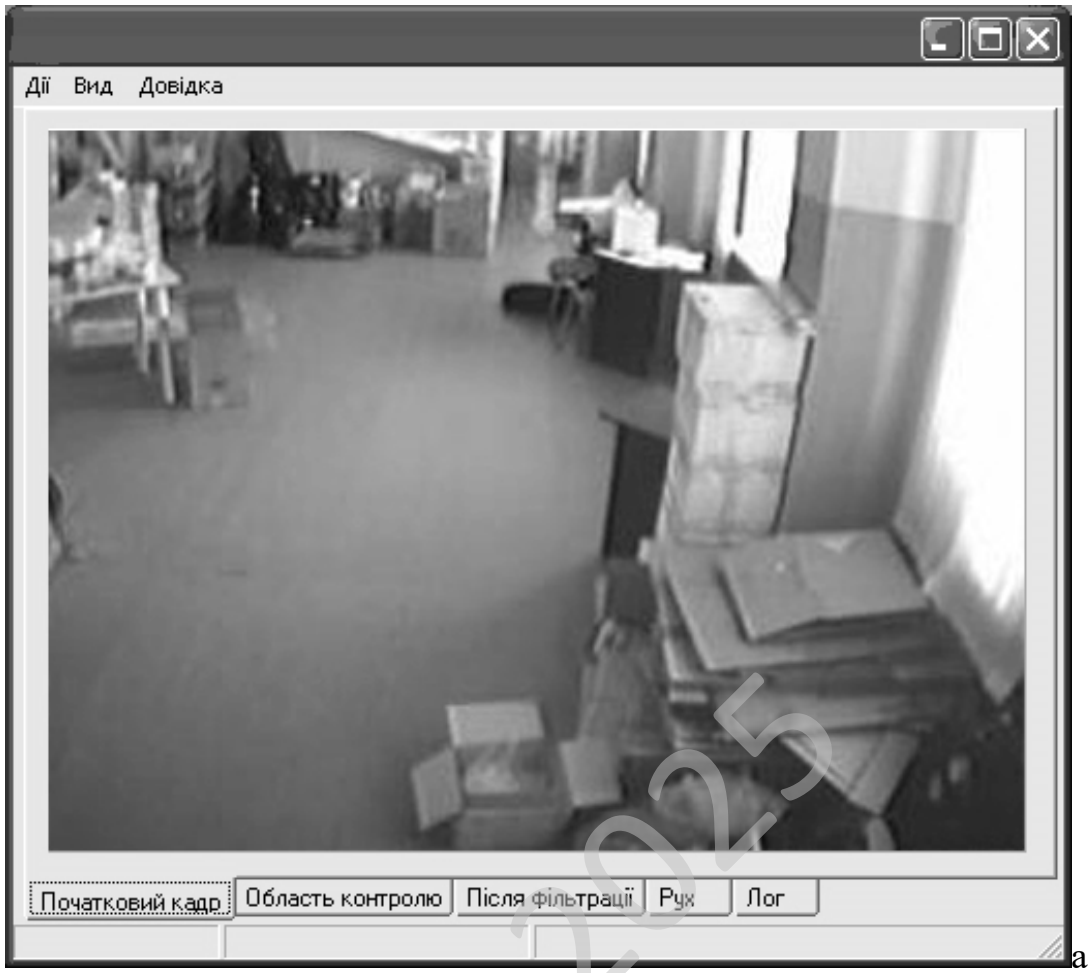
– Функціональних кнопок ПЗ.

У рамках IoT камери можуть обмінюватися даними й приймати рішення без участі посередника (центрального комп'ютера – сервера). Так, наприклад, кілька камер, об'єднаних у мережу, можуть «вісти» підозрілу особу або об'єкт: коли він виходить із сектора огляду однієї, спостереження за ним продовжує наступна. Інший можливий сценарій – узяття під контроль сусіднього сектора, коли яка-небудь камера виявляється тим або іншим способом осліплена або ушкоджена (для цього пристрій повинне бути оснащене датчиком стороннього втручання). Крім того, камера сама відправить сигнал тривоги охоронцеві, щоб він звернув увагу на сформовану ситуацію.

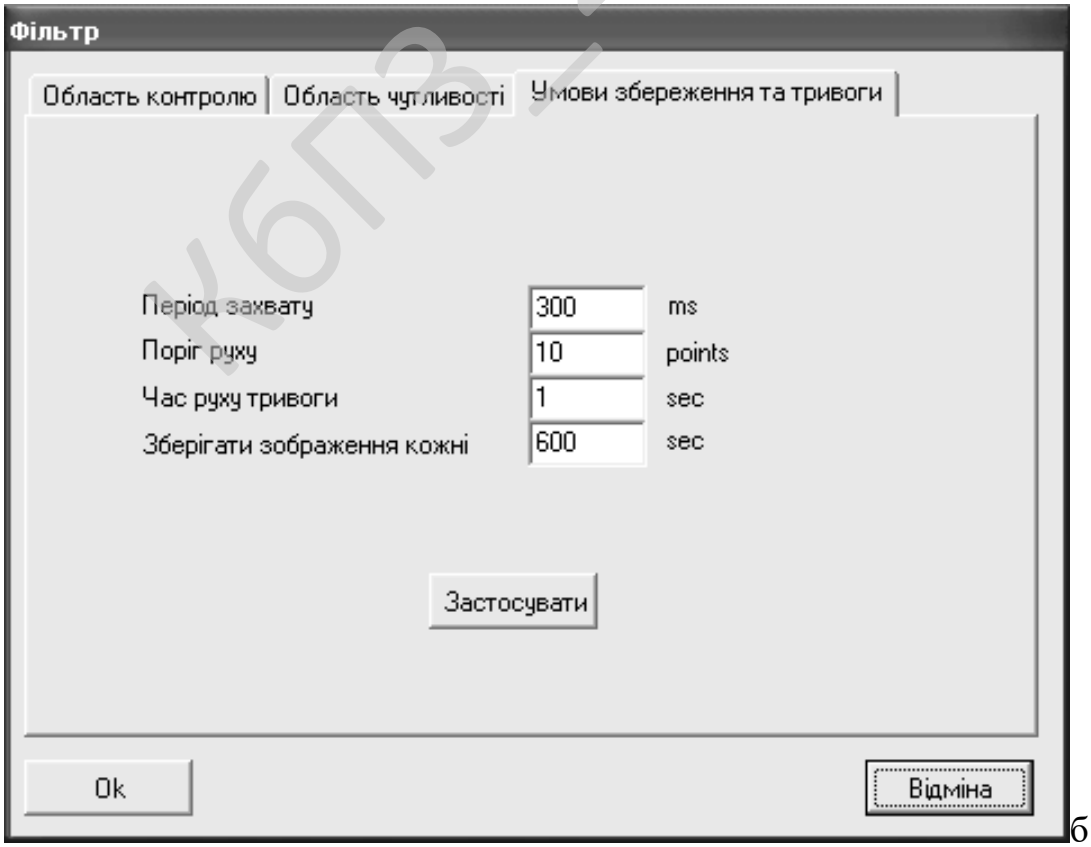
У міру того як замість колишніх аналогових пристроїв буде використовуватися усе більше пристроїв безпеки з підтримкою IP, IoT дозволить реалізувати комбіновані рішення, що поєднують колись розрізнені пристрої: камери відеоспостереження, детектори диму, пристрою доступу, гучномовці й ін. Axis випустила цілий ряд пристроїв безпеки з підтримкою IP, включаючи гучномовці, мережний відеодомофон і пристрій контролю доступу.

Взаємодія пристроїв між собою дозволить скласти більше повну картину того, що відбувається на контрольованому об'єкті.

КОПІЯ 2025



а



б

Рисунок 5.1 – Інтерфейс ПЗ: а – головне вікно, б – вікно налаштувань

КБПЗ-2025

Розроблена програма має дуже простий і інтуїтивно зрозумілий інтерфейс з користувачем. Кожен, хто в достатньому обсязі володіє операційним середовищем Windows без особливих складностей освоїть і цю програму, оскільки її інтерфейс інтуїтивно зрозумілий.

Якщо програма не видала ніяких помилок, і працює, то можна використовувати, інакше слід слідувати інструкціям, які пропонує програма.

На рисунку 5.2 зображено авторські дані розробленого програмного забезпечення.

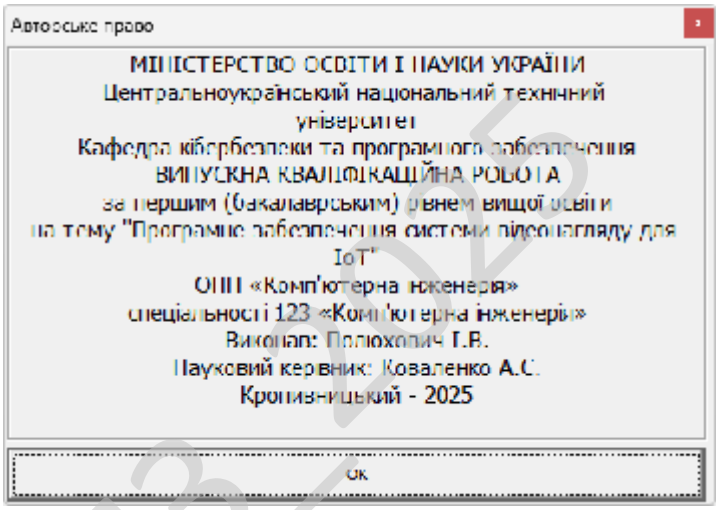


Рисунок 5.2 – Авторське право

КОПЗ-2025

Розглянемо процес впровадження програмного забезпечення, це процес налаштування програмного забезпечення під певні умови використання, а також навчання користувачів роботі з програмним продуктом. Впровадження програмного забезпечення це усі дії, що роблять розроблену програмну систему готовою до використання. Даний процес є частинною життєвого циклу програмного забезпечення.

Загалом процес розгортання складається з кількох взаємопов'язаних дій із можливими переходами між ними. Ця активність може відбуватися як з боку виробника так і з боку споживача. Оскільки кожна програмна система є

унікальною, то усі процеси та процедури під час розгортання важко передбачити. Тому, "розгортання" можна трактувати як загальний процес відповідно до певних вимог та характеристик. Розгортання може здійснюватись програмістом і в процесі розробки програмного забезпечення.

До діяльностей пов'язаних із розгортанням програмного забезпечення відносять:

– Випуск.

– Встановлення та активація.

– Деактивація.

– Адаптація.

– Обновлення.

– Вмонтування.

– Відстежування версій.

– Видалення.

– Вилучення з обігу.

При впровадженні програмного забезпечення потрібно урахувати наступні дії:

– Виділення критичних, з точки зору загального результату, процедур в діяльності організації. Коли набір таких процедур визначений, необхідно в першу чергу використовувати IT рішення для автоматизації операцій усередині саме цих процедур. Таким чином, розроблене IT рішення автоматично стає життєво важливим і затребуваним для організації, а також буде забезпечена публічність процесу впровадження;

– Розширення нормативної бази організації шляхом включення до неї регламентів, що описують порядок виконання процедур автоматизованих процесів. В іншому випадку є небезпека виникнення неузгодженості між автоматизованими процедурами та іншими процесами організації.

– Виконання робіт з загальної стандартизації існуючої діяльності організації, коли виділяються кращі практики виконання процедур і включаються

КОПЗ_2025

в ІТ рішення за принципом найбільшої корисності для більшості учасників. Відсоток таких процедур щодо загального обсягу автоматизації може бути невеликий, але це надає процесу побудови рішення вагу в організації за рахунок збільшення його необхідності.

Під час роботи над програмою було проведено тестування програмного забезпечення, тобто технічне дослідження, призначене для виявлення інформації про якість продукту відносно контексту, в якому воно має використовуватись.

Тестування включає як процес пошуку помилок або інших дефектів, так і випробування програмних складових з метою їх оцінки.

Проводилась оцінка:

– відповідності поставленим вимогам;

– правильна відповідь для усіх можливих вхідних даних;

– виконання функцій за прийнятний час;

– практичність;

– сумісність з ОС та стороннім ПЗ.

Оскільки число можливих тестів для програмних компонент практично нескінченне, тому стратегія тестування полягала в тому, щоб провести всі можливі тести з урахуванням наявного часу та ресурсів.

Як результат ПЗ тестувалось стандартним виконанням програми з метою виявлення помилок або інших дефектів.

Обрано умови розповсюдження – Shareware.

Під умовно-безплатним програмним забезпеченням можна розуміти спосіб або метод розповсюдження комерційного ПЗ на ринку (тобто на шляху до кінцевого користувача), при якому випробувачеві пропонується обмежена за можливостями (не повнофункціональна або демонстраційна версія), терміном дії (тріал версія) або версія з вбудованим набридливим нагадуванням про необхідність оплати використання програми.

КПЗ 2025

В угоді про використання (ліцензії для кінцевого користувача, EULA) також може бути обумовлена заборона на комерційне або професійне (не тестове) її використання.

Основний принцип умовно-безплатного ПЗ - «спробуй, перш ніж купити» (try before you buy). ПЗ що поширюється як умовно-безплатний, надається користувачам безоплатно. Звичайно користувач платить тільки за час завантаження файлів через Інтернет або за носій (CD диск, флешку, ключ). Протягом певного терміну, що становить зазвичай тридцять днів, він може користуватися програмою, тестувати її, освоювати її можливості.

Якщо після закінчення цього терміну користувач вирішить продовжити використання ПЗ, він зобов'язаний купити його (зареєструватися), заплативши авторові певну суму.

В іншому випадку користувач повинен припинити використання ПЗ та видалити його зі свого комп'ютера.

КбПЗ_2025

6 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти, призначено для системи відеонагляду для IoT.

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

Рішення завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем відеонагляду для IoT.

– Досліджена система відеонагляду для IoT.

– На основі отриманих результатів досліджень створена програмна реалізація системи відеонагляду для IoT.

Розроблені під час виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання відеонагляду для IoT.

Розроблене програмне забезпечення має простий, дружній та зручний інтерфейс користувача, що забезпечує легкість у освоєнні роботи програмного продукту, зручність у використанні, і не потребує особливих спеціальних знань.

При створені програмного забезпечення було використано об’єктно-орієнтований підхід, що відповідає сучасним тенденціям у галузі розробки комерційних програмних систем.

Програма реалізована на мові високого рівня Builder C++. Дана мова програмування дозволяє найбільш ефективно обробляти дані призначені для системи відеонагляду для IoT. Це дозволило мінімізувати строк розробки програмного забезпечення, і, як слід, зменшити витрати на його розробку. Запропоноване програмне забезпечення ділиться на загальне програмне забезпечення, що поставляється із засобами обчислювальної техніки й спеціальне

КПЗ - 2025

програмне забезпечення, що спеціально розроблене для даної конкретної системи й включає програми, що реалізують її функції.

Програма призначена для виконання під управлінням багатозадачної операційної системи Windows 10/11.

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

Для підвищення рівня безпеки запропоновано застосовувати алгоритм ММВ.

В цілому створене програмне забезпечення підтверджує правильність використаних проектних рішень та повністю відповідає вимогам технічного завдання. Створене програмне забезпечення має потенційну можливість для подальшого вдосконалення і застосування у різних галузях.

КбПЗ_2025

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein. Introduction to Algorithms. The MIT Press. 2022 1677 p.

2. Will Grant. 101 UX Principles. Packt Publishing. 2022. 432 p.

3. Nathan Metzler. Kotlin Programming for Beginners. Independently published. 2021. 158 p.

4. Henry Lloyd. Interactive Computer Graphics. States Academic Press. 2022. 247 p.

5. Ranjan Parekh. Fundamentals of Image, Audio, and Video Processing Using MATLAB®With Applications to Pattern Recognition. CRC Press. 2021. 406 p.

6. Alasdair McAndrew. A Computational Introduction to Digital Image Processing. Chapman & Hall. 2021. 560 p.

7. Peter Shirley, Steve Marschner. Fundamentals of Computer Graphics. 2009

8. Михайло Пічугін, Іван Канкін, Володимир Воротніков Комп'ютерна графіка. Навчальний посібник / Центр навчальної літератури 346 с. 2019р.

9. Маценко В.Г. Комп’ютерна графіка: Навчальний посібник. – Чернівці: Рута, 2009 – 343 с.

10. Інженерна комп’ютерна графіка: підручник / В.В. Проців [та ін.] / М-во освіти і науки України, Нац. гірн. унт-т. – Дніпро: НГУ, 2017. – 247 с.

11. Проців В.В. Прикладна комп'ютерна графіка [Текст]: Навч. посібник / В.В. Проців, К.А. Зіборов, К.М. Бас, Г.К. Ванжа; М-во освіти і наук, Нац. гірн. ун-т. - Д.: НГУ, 2016. - 187 с.

12. Kopf, Johannes and Lischinski, Dani. Depixelizing Pixel Art (англ.) // ACM Trans. Graph. – 2011. – Vol. 30, no. 4. – P. 99:1--99:8.

13. Giachetti, Andrea and Asuni, Nicola. Real-Time Artifact-Free Image Upscaling (англ.) // Trans. Img. Proc.. – 2011. – Vol. 20, no. 10. – P. 2760—2768.

КПЗ-2025

14. Kuznetsov, O., Frontoni, E., Kryvinska, N., Chevardin, V., Smirnov, O. «Wireless Network Encryption Stream Ciphers, Computational Modeling, and Security Analysis». *Computational Modeling and Simulation of Advanced Wireless Communication Systems*, 2024, pp. 379–402.

15. Kuznetsov, O., Frontoni, E., Kryvinska, N., Smirnov, O., Imoize, G.L. «Computational Modeling of Enhanced Spread Spectrum Codes for Asynchronous Wireless Communication». *Computational Modeling and Simulation of Advanced Wireless Communication Systems*, 2024, pp. 403–447

16. Al-Mudhafar Aqeel, A.M., Smirnova, T., Buravchenko, K., Smirnov, O. «The method of assessing and improving the user experience of subscribers in software-configured networks based on the use of machine learning». *Advanced Information Systems*, 2023, 7(2), pp. 49-56.

17. Smirnov, O., Sydorenko, V., Aleksander, M., Zhyharevych, O., Yenchev, S. «Simulation of the cloud IoT-based monitoring system for critical infrastructures». *CEUR Workshop Proceedings,* Volume 3530, 2023, pp. 256-265.

18. Smirnov, O., Odarchenko, R., Smirnova, T., Bondar, S., Volosheniuk, D. «Optimal Structure Construction of Private 5G Network for the Needs of Enterprises». Lecture Notes on Data Engineering and Communications Technologies, 2023, 178, pp. 208–223.

19. Smirnov, O., Neskorodieva, T., Fedorov, E., Rudakov, K., Neskorodieva, A. «Method Detection Audit Data Anomalies on Basis Restricted Cauchy Machine» CEUR Workshop Proceedings, Volume 3187, 2022,

20. Smirnov O., Smirnova T., Anas M. Al-Oraiqat, Drieiev O., Polishchuk L., Sheroz Khan, Yassin M. Y. Hasan, Aladdein M. Amro, Hazim S. AlRawashdeh «Method for Determining Treated Metal Surface Quality Using Computer Vision Technology». Sensors (Basel, Switzerland) Volume 22, Issue 16, 6223, 2022.

21. Smirnov O., Kuznetsov A., Kryvinska N., Kiian A., Kuznetsova K. «Full Non-Binary Constant-Weight Codes». SN Computer Science, Vol 2, 337, 2021. https://doi.org/10.1007/s42979-021-00739-w

КОПЗ–2025

22. Smirnov O., Kuznetsov A., Zhora V., Onikiychuk A., Pieshkova O. «Hiding Messages in Audio Files Using Direct Spread Spectrum». 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2021, Cracow, Poland, 22-25 September 2021. P. 414-418.

23. Smirnov O., Kuznetsov A., Lokotkova I., Kuznetsova T., Florov S., Lebid O. «Using Orthogonal Signals to Hide Information in Images». 4 IEEE International Conference on Advanced Information and Communication Technologies (AICT) - 2021, Lviv, Ukraine, September 21-25, 2021. P. 255-260.

24. Smirnov, O., Kuznetsov, A., Potii, O., Poluyanenko, N., Stelnyk, I., Mialkovsky, D. «Combining and filtering functions in the framework of nonlinear-feedback shift register». International Journal of Computing; 2020, Volume 19, Issue 2 – Research Institute for Intelligent Computer Systems – 2020. – P. 247-256.

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

26. Smirnov O., Alimseitova Zh., Adranova A., Akhmetov B., Lakhno V., Zhilkishbayeva G. «Models and algorithms for ensuring functional stability and cybersecurity of virtual cloud resources». Journal of theoretical and applied information technology Vol.98. No 21, 2020, P. 3334-3346.

27. Smirnov O., Kuznetsov A., Kovalchuk D., Kuznetsova T. «New technique for data hiding in cover images using adaptively generated pseudorandom sequences». CEUR Workshop Proceedings Volume 2654, 2020, Pages 1-14.

28. Smirnov O., Kuznetsov A., Onikiychuk A., Makushenko T., Anisimova O., Arischenko A. «Adaptive pseudo-random sequence generation for spread spectrum image steganography». 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT), Ukraine, Kyiv, May 14-18. 2020. P. 161-165.

КОПІЯ 2025

29. Smirnov O., Kuznetsov A., Kiian A., Cherep A., Kanabekova M., Chepurko I. «Testing of code-based pseudorandom number generators for post-quantum application». 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT), Ukraine, Kyiv, May 14-18. 2020. P. 172-177.

30. Smirnov O., Kuznetsov A., Pushkar'ov A., Serhiienko R., Babenko V., Kuznetsova T., «Representation of Cascade Codes in the Frequency Domain». In: Radivilova T., Ageyev D., Kryvinska N. (eds) Data-Centric Business and Applications. Lecture Notes on Data Engineering and Communications Technologies, vol 48. Springer, Cham. 2021. pp 557-587.

31. Smirnov, O., Drieieva, H., Drieiev, O., Polishchuk, Y., Brzhanov, R., Aleksander, M. «Method of fractal traffic generation by a model of generator on the graph». CEUR Workshop Proceedings Volume 2616, 2020, Pages 366-379.

32. Smirnov, O., Drieieva, H., Drieiev, O., Simakhin, V., Bondar, S., Odarchenko, R. «Managing multifractal properties of the binary sequence generated with the Markov chains», CEUR Workshop Proceedings Volume 2608, 2020, Pages 633-645.

33. Smirnov, O., Kuznetsov, A., Gorbacheva, L., Babenko, V., «Hiding data in images using a pseudo-random sequence», CEUR Workshop Proceedings Volume 2608, 2020, Pages 646-660.

34. Zhurakovskyi, B., Tsopa, N., Batrak, Y., Odarchenko, R., Smirnova, T «Comparative analysis of modern formats of lossy audio compression». Workshop Proceedings, 2020, 2654, стр. 315-327.

35. Smirnov O. Kuznetsov A., Zaicenko Yu., Pastukhov M., Oleshko O., Kuznetsova K., «Formation of Discrete Signals with Special Correlation Properties». International Conference on Information and Telecommunication Technologies and Radio Electronics, UkrMiCo 2019; Odessa; Ukraine; 9-13 September 2019. P.22-28.

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

КОПЗ_2025

Volume 18, Issue 4 – Research Institute for Intelligent Computer Systems – 2019. – P. 393-407.

37. Smirnov, O., Kuznetsov, A., Reshetniak, O., Ivko, N., Katkova, T., Kuznetsova, T., «Generators of Pseudorandom Sequence with Multilevel Function of Correlation». 2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T), Kyiv, Ukraine, 8 – 11 October 2019 . P.517-522.

38. Smirnov, O., Krasnobayev, V., Yanko, A., Kuznetsova, T. «Methods of nulling numbers in the system of residual classes». CEUR Workshop Proceedings, Vol 2588, P. 90-106, 2019.

39. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Averchev, A., Pastukhov, M., Kuznetsova, K., «Formation of Pseudorandom Sequences with Special Correlation Properties», 2019 3rd International Conference on Advanced Information and Communications Technologies, AICT -2019/ Lviv, Ukraine, 2-6 July, 2019, P. 395-399.

40. Smirnov, O., Kuznetsov, A., Kavun, S., Babenko, B., Nakisko, O., Kuznetsova, K., «Malware Correlation Monitoring in Computer Networks of Promising Smart Grids», 2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS), Kyiv, Ukraine April 17-19, 2019 P. 347-352.

41. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Pastukhov, M., Kuznetsova, K., Prokopovych-Tkachenko, D., «Discrete Signals with Special Correlation Properties», CEUR Workshop Proceedings Volume 2353, CEUR Workshop Proceedings 2019, Pages 618-629.

42. Smirnov A.A., Kuznetsov A.A., Danilenko D.A., Berezovsky A., «The statistical analysis of a network traffic for the intrusion detection and prevention systems», Telecommunications and Radio Engineering. – Volume 74, Issue 1. – Begel House Inc. – 2015. – P. 61-78.

43. Smirnov O., Kuznetsov A., Kovalchuk D., Kuznetsova T. «New Technique for Hiding Data in Cover Images Using Adaptively Generated

КБПЗ – 2025

Pseudorandom Sequences». CEUR Workshop Proceedings Volume 2732, 2020, Pages 214-227.

44. Смірнова Т.В., Конопліцька-Слободенюк О.К., Буравченко К.О., Смірнов С.А., Кравчук О.В., Козірова Н.Л., Смірнов О.А. «Дослідження технологій забезпечення кібербезпеки хмарних сервісів IaaS, PaaS та SaaS». *Кібербезпека: освіта, наука, техніка.* 2024. №4(24), С. 6-27.

45. Батрак О., Смірнова Т., Гнатюк В., Одарченко Р., Смірнов О. «Дослідження показників ефективності функціонування та перспектив розвитку систем IP-телефонії». *Підводні технології,* 2024, № 13, с. 28-35.

46. Аль-Мудхафар Акіл Абдулхуссейн М., Смірнова Т.В., Буравченко К.О., Смірнов О.А. «Метод оцінки та підвищення користувальницького досвіду абонентів в програмно-конфігурованих мережах на основі використання машинного навчання». *Сучасні інформаційні системи,* 2023, том 7, № 2, С. 49-56.

47. Т.В. Смірнова, О.М. Дрєєв, О.А. Смірнов «Хмарна інформаційна система оцінювання шорсткості з використанням дискретного частотного аналізу макрофотографій». IV міжнародна науково-практична конференція “Інформаційна безпека та комп’ютерні технології”, м. Кропивницький. 15-16 квітня 2021р. – Кропивницький: ЦНТУ. – 2021. – С. 30.

48. О.А. Смірнов, П.С. Усік, «Дослідження перспектив використання технологічних рішень в мережах 5G» у Кібербезпека та інформаційні технології: монографія. – Х. : ТОВ «ДІСА ПЛЮС», 2020.С. 122-135.

49. О.А.Смірнов, Т.В.Смірнова, Л.І. Поліщук, К.О. Буравченко, А.О.Макевнін, «Дослідження хмарних технологій як сервісів», Кібербезпека: освіта, наука, техніка. № 3(7). С. 43-62. 2020.

50. Смірнов О.А., Дрєєва Г.М., Дрєєв О.М., Смірнова Т.В. «Фрактальний аналіз генератора самоподібного трафіку на основі ланцюга Маркова». Центральноукраїнський науковий вісник. Технічні науки. № 2(33). с. 161-172, 2019.

КОПЗ_2025

51. О. Смірнов, Є. Деменко, О. Онікійчук, А. Арищенко, Л. Горбачова, «Формування псевдовипадкових послідовностей для приховування даних в зображеннях» Комп'ютерні науки та кібербезпека. № 4. С. 30-37. 2019.

52. Смірнов О.А., Конопліцька-Слободенюк О.К., Смірнов С.А., Буравченко К.О., Смірнова Т.В. Поліщук Л.І. Проектування комп'ютерних систем та мереж. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2019. – 264 с.

53. Smirnov, O., Kuznetsov, A., Kuznetsova., K. Synthesis of Discrete Signals with Improved Correlation Properties. Монографія: In.: ISCI'2019: Information Security in Critical Infrastructures. Collective monograph. Edited by Ivan D. Gorbenko and Alexandr A. Kuznetsov, ASC Academic Publishing, USA, 2019, pp. 281-299. – ISBN: 978-0-9989826-8-7 (Hardback), ISBN: 978-0-9989826-9-4 (Ebook).

54. Смірнов О.А., Дрєєва Г.М. Метод генерування фрактального трафіку за допомогою моделі генератора на графі. Монографія: Інформаційна безпека та інформаційні технології : монографія / за заг. ред. В. С. Пономаренка. – Х. : Вид. Рожко С.Г. 2019. С. 123-139

КбПЗ_2025

Додаток А
(обов’язковий)

Технічне завдання

Зміст

1 Найменування та область застосування.....2
2 Підстава для розробки.....2
3 Мета та призначення розробки.....2
4 Джерела розробки.....2
5 Технічні вимоги.....2
5.1 Вміст проекту.....2
5.2 Показники призначення.....3
5.3 Вимоги до функціональних характеристик.....3
5.4 Вимоги до архітектури.....3
5.5 Вимоги до надійності.....3
5.6 Умови експлуатації.....4
5.7 Вимоги до складу та параметрів технічних засобів.....4
5.8 Вимоги до інформаційної і програмної сумісності.....4
5.8.1 Обладнання.....4
5.8.2 Мова програмування.....4
5.8.3 Вхідні дані.....5
5.8.4 Вихідні дані.....5
6 Вимоги до програмної документації.....5
7 Перелік документів, що розробляються.....5
8 Етапи розробки.....6
9 Порядок контролю та приймання.....6

КОПЗ-2025

1 Найменування та область застосування

Це технічне завдання розповсюджується на розробку системи відеонагляду для IoT.

2 Підстава для розробки

Підставою для розробки служить завдання на випускну кваліфікаційну роботу за першим (бакалаврським) рівнем вищої освіти, видане на кафедрі кібербезпеки та програмного забезпечення (нак. № 47-02 від 17.01.2025 року).

3 Мета та призначення розробки

Метою випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти є розробка програмного забезпечення системи відеонагляду для IoT.

4 Джерела розробки

Джерелом цієї випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти є стосовна до теми література і існуючі аналоги.

5 Технічні вимоги

5.1 Склад продукції

Складниками розробки є:

– вибір і обґрунтування методів реалізації проекту;

КБПЗ_2025

– розробка програмної частин системи, а також розробка взаємодії системи з ОС та з користувачем;

– розробка програми, що реалізує спроектовані алгоритми роботи системи.

5.2 Показники призначення

Система повинна забезпечувати:

– системи відеонагляду для IoT;

– цілісність даних у процесі роботи та при зберіганні;

– простий, інтуїтивно зрозумілий інтерфейс.

5.3 Вимоги до функціональних характеристик

Розроблене програмне забезпечення не повинно мати обмежень на версію драйверів та операційної системи.

5.4 Вимоги до архітектури

Компонент, що розробляється повинен використовувати системні засоби та апаратні засоби, що на даному етапі розвитку обчислювальної техніки найбільше поширені.

5.5 Вимоги до надійності

Програмні модулі написані по всім правилам, які стосуються стандартних викликів процедур, функцій, методів і форм, визначених технічною документацією на середовище розробки.

КПЗ_2025

5.6 Умови експлуатації

Робочі місця користувачів ПЗ повинні задовольняти наступним умовам експлуатації:

– температура повітря: 19-20 град. по Цельсію;

– відносна вологість повітря до 80%;

– атмосферний тиск 107 кПа.

5.7 Вимоги до складу та параметрів технічних засобів

Програмне забезпечення повинно бути реалізоване на ПЕОМ, працювати в ОС Windows 10/11 і з сумісними з цією платформою пристроями і прикладним програмним забезпечення.

5.8 Вимоги до інформаційної і програмної сумісності

Переносність програмного забезпечення повинна бути забезпечена за рахунок його реалізації стандартного інтерфейсу взаємодії з ОС, що працюють під управлінням ОС Windows 10/11.

КбПЗ – 2025

5.8.1 Обладнання

Комп’ютер Intel® Celeron/8 Mb/1.2 Gb/SVGA 14” 1Mb або сумісні з ним.

5.8.2 Мова програмування

Середовище Builder C++.

5.8.3 Вхідні дані

Опис алгоритму роботи запропонованої системи.

5.8.4 Вихідні дані

Робоча програма.

6 Вимоги до програмної документації

Програмна продукція повинна бути представлена у виді опису структури даних, схем та опису алгоритму, а також текстів вихідних модулів програмного забезпечення згідно ЄСПД .

7 Перелік документів, що розробляються

– Структурна схема системи – 1 аркуш.
– Функціональна схема системи – 1 аркуш.
– Діаграма процесів – 1 аркуш.
– Блок-схема алгоритму роботи програми – 2 аркуша.
– Пояснювальна записка – 69 аркушів.

8 Етапи розробки

8.1 Збір і обробка інформації по темі випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти. Постановка задачі на виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти (складання ТЗ).

КБПЗ_2025

8.2 Проведення досліджень або експериментальних робіт для уточнення основних положень випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти.

8.3 Розробка функціональних схем, блок схем алгоритмів роботи програмного забезпечення.

8.4 Побудова схем взаємодії даних.

8.5 Створення прототипу ПЗ.

8.6 Віднаходження ПЗ, аналіз отриманих результатів.

8.7 Оформлення пояснювальної записки і виконання робіт по графічній частині.

9 Порядок контролю та приймання

9.1 Подання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти на попередній захист 23.05.2025 р.

9.2 Подання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти на захист 3.06.2025 р.

КбПЗ_2025

Додаток Б

(обов’язковий)

Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

ЗАТВЕРДЖУЮ

Керівник випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти

_____ Коваленко А.С.

Програмне забезпечення системи відеонагляду для IoT

КбПЗ_2025

Лістинг програми

Код документу 12

Носій: CD/DVD-диск / USB-флеш-накопичувач

Загальна кількість аркушів: 30

Літера: РП

Кропивницький – 2025 року

// MainUnit.cpp – основна програма системи відеонагляду для IoT

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "MainUnit.h"
#include "SetDeviceUnit.h"
#include "SetFilterUnit.h"
#include "SetCommonUnit.h"
#include "ReConst.hpp"
#include "reinit.hpp"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma link "trayicon"
#pragma resource "*.dfm"
TMainForm *MainForm;
DevSettings Sets;
bool bWork = false;
AnsiString ConfigName;
//---------------------------------------------------------------------------

__fastcall TMainForm::TMainForm(TComponent* Owner)
  : TForm(Owner),
    bWork(false),
    LastSaveTime(0),
    LastFrameTime(0)
{
   // Формуємо ім'я файлу з налаштуваннями
   ConfigName = ExtractFilePath(Application->ExeName) + "va.cfg";

   // Створюємо TBitmap для обробки кадрів
   CaptureBitmap = new Graphics::TBitmap;

   // Налаштовуємо таймер (переконайтеся, що CaptureTimer додано на форму)
   CaptureTimer->Enabled = false;
   CaptureTimer->Interval = Sets.CycleTime;  // Початковий інтервал
}

//---------------------------------------------------------------------------
__fastcall TMainForm::~TMainForm()
{
   delete CaptureBitmap;
}

//---------------------------------------------------------------------------
void __fastcall TMainForm::ActivateButtonClick(TObject *Sender)
{
   // Перемикаємо робочий режим
   bWork = !bWork;
   if(bWork)
   {
      // Якщо не вдалося підключитися — скидаємо режим
      if (!Sets.Connect())
      {
         bWork = false;
         return;
      }
      ActivateButton->Caption = "Стоп";
      LastSaveTime  = GetTickCount();
      LastFrameTime = LastSaveTime;
      CaptureTimer->Interval = Sets.CycleTime;
      CaptureTimer->Enabled = true;
   }
   else
   {
```

КПЗ_2025

```
      ActivateButton->Caption = "Активувати";
      CaptureTimer->Enabled = false;
      Sets.Disconnect();
   }
}

//---------------------------------------------------------------------------
void __fastcall TMainForm::CaptureTimerTimer(TObject *Sender)
{
   DWORD currentTime = GetTickCount();
   DWORD frameTime   = currentTime - LastFrameTime;
   LastFrameTime     = currentTime;

   // Отримуємо кадр з пристрою
   capGrabFrame(Sets.hCaptureW);

   // Відтворюємо обробку кадру в залежності від вибраної вкладки
   switch (MainPageControl->TabIndex)
   {
      case 0: // Вхідний кадр
         PrevPB->Canvas->StretchDraw(PrevPB->ClientRect, Sets.FramePic->Bitmap);
         break;
      case 1: // Контрольний кадр
         Sets.PaintArray(Sets.GrayData, CaptureBitmap);
         FramePB->Canvas->StretchDraw(FramePB->ClientRect, CaptureBitmap);
         break;
      case 2: // Відфільтрований кадр
         Sets.PaintArray(Sets.FiltData, CaptureBitmap);
         FiltPB->Canvas->StretchDraw(FiltPB->ClientRect, CaptureBitmap);
         break;
      case 3: // Кадр із рухом
         Sets.PaintArray(Sets.DifData, CaptureBitmap);
         MotionPB->Canvas->StretchDraw(MotionPB->ClientRect, CaptureBitmap);
         break;
      default:
         break;
   }

   // Автоматичне збереження кадру, якщо пройшло достатньо часу
   if (currentTime - LastSaveTime > Sets.ForceSaveTime * 1000)
   {
      LastSaveTime = currentTime;
      Sets.SaveFrame();
   }

   // Оновлюємо статус-бар
   StatusBar->Panels->Items[0]->Text =
      "FPS: " + IntToStr(1000 / (frameTime == 0 ? 1 : frameTime)) +
      " (" + IntToStr(frameTime) + "ms)";
   StatusBar->Panels->Items[1]->Text =
      "Кадрів оброблено: " + IntToStr(Sets.FramesCaptured);
   StatusBar->Panels->Items[2]->Text =
      "Рухомих точок: " + IntToStr(Sets.ActivePoints);
}

//---------------------------------------------------------------------------
void __fastcall TMainForm::DeviceItemClick(TObject *Sender)
{
   if(!Sets.bConnected)
      Sets.Connect();

   SetDeviceForm->ShowModal();
   // Можна повторно підключитися, якщо потрібно
   Sets.Connect();
}

//---------------------------------------------------------------------------
void __fastcall TMainForm::FormShow(TObject *Sender)
{
```

КОПЗ_2025

```
   Sets.hParentW = CaptPanel->Handle;
   Sets.LogMemo  = LogMemo;
   Sets.LoadFromFile(ConfigName);

   AnsiString asLcid = LoadStr(_SLcid.id);
   int iLcid = asLcid.IsEmpty() ? 0 : asLcid.ToInt();

   if ((iLcid != Sets.LocalId) &&
       (Sets.LocalId == ENGLISH || Sets.LocalId == RUSSIAN))
   {
      if (LoadNewResourceModule(Sets.LocalId) != 0)
         ReinitializeForms();
   }
   asLcid = LoadStr(_SLcid.id);
   iLcid = asLcid.IsEmpty() ? 0 : asLcid.ToInt();
   EnglishItem->Checked = (iLcid == EnglishItem->Tag);
   RussianItem->Checked = (iLcid == RussianItem->Tag);

   if (!Sets.bConnected && Sets.Connect())
      capGrabFrame(Sets.hCaptureW);
}
}

// StatusCallbackProc: стан функції зворотнього виклику
// hWnd:               дескриптор вікна захвату
// nID:                код стану для поточного стану
// lpStatusText:       рядок повідомлення для поточного стану
//
LRESULT PASCAL DevSettings::StatusCallbackProc(HWND hWnd, int nID,
    LPSTR lpStatusText)
{

    if (nID == 0) {           // Видалити старі повідомлення про стан
        return (LRESULT) TRUE;
    }
    // Показати ID стану та текстове повідомлення про стан
    Sets.AddToLog("Status("+IntToStr(nID)+"): "+lpStatusText);
    return (LRESULT) TRUE;
}


// ErrorCallbackProc: помилка функції зворотнього виклику
// hWnd:              дескриптор вікна захвату
// nErrID:            код помилки для неочікуваної помилки
// lpErrorText:       повідомлення про помилку для неочікуваної помилки
//
LRESULT PASCAL DevSettings::ErrorCallbackProc(HWND hWnd, int nErrID,
    LPSTR lpErrorText)
{
    if (nErrID == 0)            // Початок нової головної функції
        return (LRESULT)TRUE;            // Очистити старі помилки

    // Показати ідентифікатор помилки та текст
    Sets.AddToLog("Помилка("+IntToStr(nErrID)+"): "+lpErrorText);
    MessageBox(hWnd, lpErrorText, AnsiString("Помилка
#"+IntToStr(nErrID)).c_str(), MB_OK | MB_ICONEXCLAMATION);
    return (LRESULT) TRUE;
}



// FrameCallbackProc: функція зворотнього виклику кадру
// hWnd:              дескриптор вікна захвату
// lpVHdr:            вказівник на структуру, яка містить
//                    інформацію про кадр
//
LRESULT PASCAL DevSettings::FrameCallbackProc(HWND hWnd, LPVIDEOHDR lpVHdr)
{
    Sets.UpdateFrame(lpVHdr);
```

КПЗ 2025

```
    return (LRESULT) TRUE ;
}


bool DevSettings::Connect(){
   int nID = 0;
   Disconnect();
   if(!hCaptureW) hCaptureW = capCreateCaptureWindow("Система
відеоспостереження",
                              WS_CHILD,    0, 0, 100, 100, hParentW, nID);
   bool fOK = false;
   if(hCaptureW && DevIndex != -1){
     //установка функції зворотнього виклику
     fOK = capSetCallbackOnError(hCaptureW, &DevSettings::ErrorCallbackProc);
     if(fOK) fOK = capSetCallbackOnStatus(hCaptureW,
&DevSettings::StatusCallbackProc);
     if(fOK) fOK = capSetCallbackOnFrame(hCaptureW,
&DevSettings::FrameCallbackProc);
     //під'єднатися до драйвера
     if(fOK) fOK = capDriverConnect(hCaptureW, DevIndex);
     //установка параметрів захвату
     if(fOK && CaptureParms.dwRequestMicroSecPerFrame > 0){
        fOK = capCaptureSetSetup(hCaptureW, &CaptureParms, sizeof
(CAPTUREPARMS));
        if(!fOK) fOK = capCaptureGetSetup(hCaptureW, &CaptureParms,
sizeof(CAPTUREPARMS));
     }else if(fOK) fOK = capCaptureGetSetup(hCaptureW, &CaptureParms,
sizeof(CAPTUREPARMS));
     //установка формату відео
     if(fOK && VideoFormat){
        fOK = capSetVideoFormat(hCaptureW, VideoFormat, VideoFormatSize);
     }
     if((!fOK && VideoFormat) || (fOK && !VideoFormat)){
        VideoFormatSize = capGetVideoFormatSize(hCaptureW);
        if(VideoFormat){
           delete[] VideoFormat;
           VideoFormat = NULL;
        }
        if(VideoFormatSize > 0){
           VideoFormat = (LPBITMAPINFO) new BYTE[VideoFormatSize];
           fOK = capGetVideoFormat(hCaptureW, VideoFormat, VideoFormatSize);
        }
     }
     if(fOK){
        //установка розміру вікна
        capGetStatus(hCaptureW, &CapStatus, sizeof (CAPSTATUS));
       // SetWindowPos(hCaptureW, NULL, 0, 0, CapStatus.uiImageWidth,
       //              CapStatus.uiImageHeight, SWP_NOZORDER | SWP_NOMOVE);
        FramePic->Bitmap->Width = CapStatus.uiImageWidth;
        FramePic->Bitmap->Height = CapStatus.uiImageHeight;
        FramePic->Bitmap->HandleType = bmDIB;
        FramePic->Bitmap->PixelFormat = pf24bit;
        FramePic->Bitmap->IgnorePalette = true;

        GrayPic->Bitmap->Width = pW;
        GrayPic->Bitmap->Height = pH;
        GrayPic->Bitmap->HandleType = bmDIB;
        GrayPic->Bitmap->PixelFormat = pf24bit;
        GrayPic->Bitmap->IgnorePalette = true;

        if(GrayData) delete[] GrayData;
        GrayData = new BYTE[pW * pH];
        memset(GrayData, 0, pW*pH);

        //if(FiltData) delete[] FiltData;
        //FiltData = new BYTE[pW * pH];
     }

   }
```

КоПЗ_2025

```
   if(!fOK){
      AddToLog("Помилка: неможливо під'єднатися!");
      bConnected = false;
   }else{
      bConnected = true;
      AddToLog("Стан: Драйвер захвату під'єднано");
   }
   CurBufFrame = 0;
   FramesCaptured = 0;
   return fOK;
}



bool DevSettings::Disconnect(){
   if(!bConnected && !hCaptureW) return true;
   if(hCaptureW){
     // вивільнити функції зворотнього виклику
     capSetCallbackOnError(hCaptureW, NULL);
     capSetCallbackOnStatus(hCaptureW, NULL);
     capSetCallbackOnVideoStream(hCaptureW, NULL);
     capSetCallbackOnFrame(hCaptureW, NULL);
     //від'єднатися від пристрою
     capDriverDisconnect(hCaptureW);
     DestroyWindow(hCaptureW);
     hCaptureW = NULL;
   }
   bConnected = false;
   AddToLog("Стан: Драйвер захвату від'єднано");
   return true;
}

void DevSettings::UpdateFrame(LPVIDEOHDR lpVHdr)
{
   HDC hdc;
   HDRAWDIB hdd;
   hdd = DrawDibOpen();

   //виводити вихідний кадр в FramePic
   hdc = FramePic->Bitmap->Canvas->Handle ;
   FramePic->Bitmap->Canvas->TryLock();
   DrawDibBegin(hdd, hdc,
                VideoFormat->bmiHeader.biWidth, //dxDest
                VideoFormat->bmiHeader.biHeight, //dyDest
                &VideoFormat->bmiHeader, //lpbi
                VideoFormat->bmiHeader.biWidth, //dxSrc
                VideoFormat->bmiHeader.biHeight, //dySrc
                NULL);
   DrawDibDraw(hdd, hdc,
               0, //xDst
               0, //yDst
               VideoFormat->bmiHeader.biWidth, //dxDst
               VideoFormat->bmiHeader.biHeight, //dyDst
               &VideoFormat->bmiHeader, //lpbi
               lpVHdr->lpData, //lpBits
               0, //xSrc
               0, //ySrc
               VideoFormat->bmiHeader.biWidth, //dxSrc
               VideoFormat->bmiHeader.biHeight, //dySrc
               NULL);
   DrawDibEnd(hdd);
   FramePic->Bitmap->Canvas->Unlock();


   //виводити вихідний кадр в GrayPic (для обробки)
   DWORD xSrc = 0, ySrc = 0,
         dxSrc = VideoFormat->bmiHeader.biWidth,
         dySrc = VideoFormat->bmiHeader.biHeight;
   if(ClipRect.bottom > 0){
```

КПЗ_2025

```
      //використати відсікання
      float MX = dxSrc / (float)pW,
            MY = dySrc / (float)pH;
      xSrc = ClipRect.left * MX;
      ySrc = ClipRect.top * MX;
      dxSrc = (ClipRect.right - ClipRect.left) * MX;
      if(dxSrc > (DWORD)VideoFormat->bmiHeader.biWidth - xSrc) dxSrc =
VideoFormat->bmiHeader.biWidth - xSrc;
      dySrc = (ClipRect.bottom - ClipRect.top) * MY;
      if(dySrc > (DWORD)VideoFormat->bmiHeader.biHeight - ySrc) dySrc =
VideoFormat->bmiHeader.biHeight - ySrc;
   }
   hdc = GrayPic->Bitmap->Canvas->Handle ;
   GrayPic->Bitmap->Canvas->TryLock();
   DrawDibBegin(hdd, hdc,
                pW, //dxDest
                pH, //dyDest
                &VideoFormat->bmiHeader, //lpbi
                dxSrc,
                dySrc,
                NULL);
   //DrawDibRealize(hdd, hdc, fBackground);
   DrawDibDraw(hdd, hdc,
               0, //xDst
               0, //yDst
               pW, //dxDst
               pH, //dyDst
               &VideoFormat->bmiHeader, //lpbi
               lpVHdr->lpData, //lpBits
               xSrc,
               ySrc,
               dxSrc,
               dySrc,
               NULL);
   DrawDibEnd(hdd);
   DrawDibClose(hdd);
   GrayPic->Bitmap->Canvas->Unlock();

   // створити зображення у відтінках сірого і автоматично збалансувати його
   BITMAP bm;
   GetObject(GrayPic->Bitmap->Handle,sizeof(BITMAP),&bm);
   register DWORD offs;
   DWORD maxoffs = bm.bmHeight * bm.bmWidthBytes ;
   register int Y;
   BYTE minY = 255, maxY = 0;
   for(offs = 0; offs < maxoffs; offs+= 3){
         Y =
((int)((BYTE*)bm.bmBits)[offs]+(int)((BYTE*)bm.bmBits)[offs+1]+(int)((BYTE*)bm.b
mBits)[offs+2]) / 3;
         if(minY > Y) minY = Y;
         if(maxY < Y) maxY = Y;
         GrayData[offs/3] = Y;
   }

   maxoffs = maxoffs/3;
   if(maxY - minY == 0){
     AddToLog(AnsiString("Помилка захвату зображення ")+__LINE__+" "+__FILE__);
     return;
   }
   static float M = 255/(maxY - minY);
   static float avminY = minY, avmaxY = maxY;
   avmaxY = 0.95*avmaxY + 0.05*maxY;
   if(avmaxY > 255) avmaxY = 255;

   avminY = 0.95*avminY + 0.05*minY;
   if(avminY > 250) avminY = 250;

   if(avmaxY <= avminY) avmaxY = avminY + 1;
   M = 255/(avmaxY - avminY);
```

КОПІЯ_2025

```
   for(offs = 0; offs < maxoffs; offs++){
      Y = ((int)GrayData[offs] - avminY) * M ;
      if(Y < 0) GrayData[offs] = 0;
      else if(Y > 255) GrayData[offs] = 255;
      else GrayData[offs] = Y;
   }


   CurBufFrame++;
   CurBufFrame = CurBufFrame % (FrameBufCnt-1);

   FiltData = (BYTE*)FrameBuffer->Items[CurBufFrame];
   FilterFrame(GrayData, FiltData);
   FramesCaptured ++;

   if(FramesCaptured > 2){
      CompareFrames(FiltData,
                    (BYTE*)FrameBuffer->Items[CalcBufIndex(CurBufFrame-1)],
                    DifData );
   }

}



void __fastcall TMainForm::FormClose(TObject *Sender, TCloseAction &Action)
{
  bWork = false;
  Sets.SaveToFile(ConfigName);
  Sets.Disconnect();
}
//---------------------------------------------------------------------------


void __fastcall TMainForm::FilterItemClick(TObject *Sender)
{
   if(!Sets.bConnected) Sets.Connect();
   if(Sets.bConnected) capGrabFrame(Sets.hCaptureW);
   SetFilterForm->ShowModal();
}
//---------------------------------------------------------------------------


void DevSettings::FilterFrame(BYTE* Src, BYTE* Dst)
{
   register int idx, idy, x, y, i, offs, toffs, cnt;
   int ytop, ybot, dx, dy, srw, srh;
   float SY;
   DWORD sum;
   SensRegion *sr = (SensRegion*)SensList->Items[0];
   srw = sr->right - sr->left;
   srh = sr->bottom - sr->top;

   for(i=0; i<SensList->Count; i++){
      sr = (SensRegion*)SensList->Items[i];
      ybot = pH-sr->top; ytop = pH-sr->bottom;
      dx = (100 - sr->sens) * (srw)/400;
      dy = (100 - sr->sens) * (srh)/300;
      SY =  sr->sens/100.0*0.4 + 0.6;
      //dx=0; dy=5;
      for(y = ytop; y < ybot; y++){
         for(x = sr->left; x < sr->right; x++){
            offs = pW*y + x;
            if(sr->sens == 0){
               Dst[offs] = 0;
            }else if(sr->sens == 100){
               Dst[offs] = Src[offs];
            }
            else{
```

К6ТВ_2025

```
                sum = 0;
                cnt = 0;
                for(idy = y-dy; idy <= y+dy; idy++){
                   for(idx = x-dx; idx <= x+dx; idx++){
                      toffs = pW*((idy < ytop)? idy+srh : (idy >= ybot)? idy-srh
: idy) +
                              ((idx < sr->left)? idx+srw : ((idx >= sr->right)?
idx-srw : idx));
                    //  toffs = pW*((idy < 0)? idy+srh : (idy >= pH)? idy-srh :
idy) +
                    //          ((idx < 0)? idx+srw : ((idx >= pW)? idx-srw :
idx));
                      sum+= Src[toffs];
                      cnt++;
                   }
                }
                Dst[offs] = sum / cnt * SY;
             }
          }
       }
    }
}

void DevSettings::PaintArray(BYTE * arr, Graphics::TBitmap * bmp)
{
   BITMAP bm;
   bmp->Width = pW;
   bmp->Height = pH;
   bmp->PixelFormat = pf24bit;
   GetObject(bmp->Handle, sizeof(BITMAP), &bm);
   DWORD maxoffs = bm.bmHeight * bm.bmWidthBytes,
         offs;
   BYTE Y;
   if(!arr) return;
   for(offs = 0; offs < maxoffs; offs+= 3){
         Y = arr[offs/3];
         ((BYTE*)bm.bmBits)[offs] = Y;
         ((BYTE*)bm.bmBits)[offs+1] = Y;
         ((BYTE*)bm.bmBits)[offs+2] = Y;
   }
}



void DevSettings::CompareFrames(BYTE * f1, BYTE * f2, BYTE * dest)
{
    register int offs, maxoffs = pW * pH, Y;
    static int StartMotionTime;
    ActivePoints = 0;
    for(offs = 0; offs < maxoffs; offs++){
       Y = abs(f2[offs] - f1[offs]) * 5;
       if(Y >= 255){
          Y = 255;
          ActivePoints ++;
       }
       dest[offs] = Y;
    }

    if(ActivePoints >= MaxActivePoints){
       if(MotionTime == 0) StartMotionTime = GetTickCount();
       MotionTime = GetTickCount() - StartMotionTime + 1;
       SaveFrame();
       if(MotionTime > AlarmMotionTime * 1000){
          DoAlarm();
          MotionTime = 0;
       }
    }else{
       MotionTime = 0;
```

K6IR_2025

```
    }

}


int inline DevSettings::CalcBufIndex(int arg)
{
   return ((arg >= FrameBufCnt-1)? arg%(FrameBufCnt-1) : ((arg < 0)? arg +
FrameBufCnt - 1 : arg));
}


void __fastcall TMainForm::PrevPBPaint(TObject *Sender)
{
   Graphics::TBitmap *tbmp = new Graphics::TBitmap;

   switch (MainPageControl->TabIndex){
      case 0: //вхідний кадр
         PrevPB->Canvas->StretchDraw(PrevPB->ClientRect, Sets.FramePic->Bitmap);
         break;
      case 1: //контрольний кадр
         Sets.PaintArray(Sets.GrayData, tbmp);
         FramePB->Canvas->StretchDraw(FramePB->ClientRect, tbmp);
         break;
      case 2: //відфільтрований кадр
         Sets.PaintArray(Sets.FiltData, tbmp);
         FiltPB->Canvas->StretchDraw(FiltPB->ClientRect, tbmp);
         break;
      case 3: //кадр із рухом
         Sets.PaintArray(Sets.DifData, tbmp);
         MotionPB->Canvas->StretchDraw(MotionPB->ClientRect, tbmp);
         break;
   }

   delete tbmp;
}
//---------------------------------------------------------------------------

void __fastcall TMainForm::SaveItemClick(TObject *Sender)
{
   Sets.SaveToFile(ConfigName);
}
//---------------------------------------------------------------------------



void __fastcall TMainForm::LogMemoChange(TObject *Sender)
{
   if(LogMemo->Lines->Count > 0)
      StatusBar->Panels->Items[3]->Text = LogMemo->Lines->Strings[LogMemo-
>Lines->Count-1];
}
//---------------------------------------------------------------------------

void __fastcall TMainForm::Show1Click(TObject *Sender)
{
   Application->Restore();
}
//---------------------------------------------------------------------------

void __fastcall TMainForm::Hide1Click(TObject *Sender)
{
   Application->Minimize();
}
//---------------------------------------------------------------------------


void DevSettings::DoAlarm()
{
```

КбПЗ_2025

```
    AddToLog("ТРИВОГА!");

    if(AlarmExecuteFN != "" && FileExists(AlarmExecuteFN)){
       if(ExtractFileExt(AlarmExecuteFN).LowerCase() == ".wav"){
          PlaySound(AlarmExecuteFN.c_str(), NULL, SND_ASYNC|SND_FILENAME);
       }else{
          ShellExecute(
                       hParentW,    // дескриптор батьківського вікна
                       "open",      // вказівник на рядок, який визначає
операцію для виконання
                       AlarmExecuteFN.c_str(), // вказівник  на рядок із іменем
файлу або папки
                       NULL,  // вказівник на рядок, який вихначає параметри
виконуваних файлів
                       NULL,  // вказівник на рядок, який визначає папку по
замовчуванню
                       SW_SHOWDEFAULT          // чи показується файл, коли його
відкрили
                       );
       }
    }

}


void DevSettings::SaveFrame()
{
  AnsiString FN = EvalImgPath();
  if(!DirectoryExists(ExtractFilePath(FN)))
ForceDirectories(ExtractFilePath(FN));

  FramePic->Bitmap->Canvas->TextOut(0, 0, DateTimeToStr(Now()));

  TJPEGImage *jp = new TJPEGImage();
  try
  {
    jp->Assign(FramePic->Bitmap);
    jp->CompressionQuality = JPEGCompression;
    jp->Compress();
    jp->SaveToFile(FN);
  }
  catch(...){
    delete jp;
    return;
  }
  delete jp;
  AddToLog("Кадр збережено");
}

AnsiString DevSettings::EvalImgPath(){
  static unsigned int sn = 0;
  sn++ ;
  TDateTime dt = Now();
  AnsiString tS, res = FNTemplate;
  AnsiString dd, ddd, mm, mmm, yy, yyyy, hh, nn, ss, zzz;
  DateTimeToString(dd, "dd", dt);
  DateTimeToString(ddd, "ddd", dt);
  DateTimeToString(mm, "mm", dt);
  DateTimeToString(mmm, "mmm", dt);
  DateTimeToString(yy, "yy", dt);
  DateTimeToString(yyyy, "yyyy", dt);
  DateTimeToString(hh, "hh", dt);
  DateTimeToString(nn, "nn", dt);
  DateTimeToString(ss, "ss", dt);
  DateTimeToString(zzz, "zzz", dt);
  DateTimeToString(tS, "dd_mmm_yyyy", dt);  // {dd}_{mmm}_{yyyy}\\{dd}{mmm}{yy}-
{hh}{nn}{ss}{zzz}.jpg
  AnsiString FN = ImagesFolder + "\\" + tS + "\\";
  //if(!DirectoryExists(FN)) CreateDir(FN);
```

КбПЗ_2025

```
  DateTimeToString(tS, "ddmmmyy-hhnnsszzz", Now());
  FN = FN + tS + ".jpg";
  TReplaceFlags rf; rf << rfReplaceAll;
  res = StringReplace(res, "{dd}", dd, rf);
  res = StringReplace(res, "{ddd}", ddd, rf);
  res = StringReplace(res, "{mm}", mm, rf);
  res = StringReplace(res, "{mmm}", mmm, rf);
  res = StringReplace(res, "{yy}", yy, rf);
  res = StringReplace(res, "{yyyy}", yyyy, rf);
  res = StringReplace(res, "{hh}", hh, rf);
  res = StringReplace(res, "{nn}", nn, rf);
  res = StringReplace(res, "{ss}", ss, rf);
  res = StringReplace(res, "{zzz}", zzz, rf);
  res = StringReplace(res, "{sn}", IntToStr(sn), rf);
  return ImagesFolder + "\\" + res + ".jpg";
}

void __fastcall TMainForm::CommonItemClick(TObject *Sender)
{
   SetCommonForm->ShowModal();
}
//---------------------------------------------------------------------------

void __fastcall TMainForm::OpenImgFolderBitBtnClick(TObject *Sender)
{
    ShellExecute(
                Handle,      // дескриптор батьківського вікна
                "open",      // вказівник на рядок, який визначає операцію для
виконання
                "explorer.exe",    // вказівник  на рядок із іменем файлу або
папки
                Sets.ImagesFolder.c_str(),     // вказівник на рядок, який
вихначає параметри виконуваних файлів
                NULL,  // вказівник на рядок, який визначає папку по
замовчуванню
                SW_SHOWDEFAULT          // чи показується файл, коли його
відкрили
                );
}
//---------------------------------------------------------------------------

void __fastcall TMainForm::ShootNowBitBtnClick(TObject *Sender)
{
   if(!Sets.bConnected) Sets.Connect();
   if(Sets.bConnected){
      capGrabFrame(Sets.hCaptureW);
      Sets.SaveFrame();
      PrevPBPaint(Sender);

   }
}
//---------------------------------------------------------------------------

void __fastcall TMainForm::FormCreate(TObject *Sender)
{
  EnglishItem->Tag = ENGLISH;
  RussianItem->Tag = RUSSIAN;

  AnsiString asLcid;
  int iLcid = 0;
  asLcid = LoadStr(_SLcid.id);
  if(asLcid != "") iLcid = asLcid.ToInt();

  EnglishItem->Checked = iLcid == EnglishItem->Tag;
  RussianItem->Checked = iLcid == RussianItem->Tag;
}
//---------------------------------------------------------------------------

void __fastcall TMainForm::EnglishItemClick(TObject *Sender)
```

КПЗ – 2025

```
{
  if(LoadNewResourceModule(dynamic_cast<TComponent*>(Sender)->Tag) != 0)
  {
    bWork = false;
    Sets.Disconnect();
    Application->ProcessMessages();
    Sets.LocalId = dynamic_cast<TComponent*>(Sender)->Tag;
    Sets.SaveToFile(ConfigName);
    ReinitializeForms();
    EnglishItem->Checked = EnglishItem == Sender;
    RussianItem->Checked  = RussianItem == Sender;
  }

}
//---------------------------------------------------------------------------
```

КбПЗ_2025

// MainUnit.h – заголовочний файл основної програми

```
//---------------------------------------------------------------------------

#ifndef MainUnitH
#define MainUnitH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <Menus.hpp>
#include "vfw.h"
#include <ExtCtrls.hpp>
#include <ComCtrls.hpp>
#include <Graphics.hpp>
#include <Buttons.hpp>
#include <jpeg.hpp>
#include <ActnList.hpp>
#include <ExtActns.hpp>
//---------------------------------------------------------------------------
struct SensRegion:public RECT{
   int sens;
};

class DevSettings{
  public:
    HWND hParentW,
         hCaptureW;
    TMemo *LogMemo;
    TPicture *FramePic;
    TPicture *GrayPic;
    TList *SensList;
    TList *FrameBuffer;
    BYTE *GrayData,
         *FiltData, *DifData;
    int pW, pH;
    int SensRectNx, SensRectNy;
    int FrameBufCnt;
    DWORD FramesCaptured;
    int ActivePoints;
    int CurBufFrame;
    int DevIndex;
    bool bConnected;
    int CycleTime;
    int MaxActivePoints;
    int AlarmMotionTime;
    int ForceSaveTime;
    int MotionTime;
    AnsiString AlarmExecuteFN;
    AnsiString ImagesFolder;
    AnsiString FNTemplate;
    int JPEGCompression;
    int LocalId;

    CAPDRIVERCAPS CapDrvCaps;
    CAPSTATUS CapStatus;
    CAPTUREPARMS CaptureParms;
    LPBITMAPINFO VideoFormat;
    DWORD VideoFormatSize;
    RECT ClipRect;

    DevSettings(){
       JPEGCompression = 70;
       ImagesFolder = ".";
       FNTemplate = "{dd}_{mm}_{yyyy}\\{dd}{mmm}{yy}-{hh}{nn}{ss}{zzz}";
       MotionTime = 0;      //  мс
       CycleTime = 300;     //  мс
```

КОТЗ_2025

```
        MaxActivePoints = 10;
        AlarmMotionTime = 1; // в секундах
        ForceSaveTime = 600; // секунди
        DevIndex = -1;
        VideoFormatSize = 0;
        pW = 320; pH = 240;
        SensRectNx = SensRectNy = 8;
        ClipRect.bottom = 0;
        FrameBufCnt = 10;
        ActivePoints = 0;
        bConnected = false;
        hParentW = hCaptureW = VideoFormat = NULL;
        LogMemo = NULL;
        GrayData = FiltData = NULL;
        FramePic = new TPicture;
        GrayPic = new TPicture;
        SensList = new TList;
        FrameBuffer = new TList;
        int nx, ny, dx, dy;
        SensRegion *sR;
        dx = pW / SensRectNx;
        dy = pH / SensRectNy;
        for(nx = 0; nx < SensRectNx; nx++){
           for(ny = 0; ny < SensRectNy; ny++){
              sR = new SensRegion;
              sR->left = nx * dx;
              sR->right = sR->left + dx;
              sR->top = ny*dy;
              sR->bottom = sR->top + dy;
              sR->sens = 100;
              SensList->Add(sR);
           }
        }
        BYTE *tbuf;
        for(nx = 0; nx < FrameBufCnt; nx++){
           tbuf = new BYTE[pW*pH];
           memset(tbuf, 0, pW*pH);
           FrameBuffer->Add(tbuf);
        }
        DifData = (BYTE*)FrameBuffer->Items[FrameBufCnt - 1];
    }
    ~DevSettings(){
        if(VideoFormat) delete[] VideoFormat;
        if(GrayData) delete[] GrayData;
        //if(FiltData) delete[] FiltData;
        delete FramePic;
        delete GrayPic;
        for(int i = 0; i < SensList->Count; i++){
          delete (SensRegion*)(SensList->Items[i]);
        }
        delete SensList;

        for(int i = 0; i < FrameBuffer->Count; i++){
          delete (BYTE*)(FrameBuffer->Items[i]);
        }
        delete FrameBuffer;
    }
    void SaveToFile(AnsiString FN);
    void LoadFromFile(AnsiString FN);
    bool Connect();
    bool Disconnect();
    //callback functions
    static LRESULT PASCAL StatusCallbackProc(HWND hWnd, int nID, LPSTR
lpStatusText);
    static LRESULT PASCAL ErrorCallbackProc(HWND hWnd, int nErrID, LPSTR
lpErrorText);
    static LRESULT PASCAL FrameCallbackProc(HWND hWnd, LPVIDEOHDR lpVHdr);
    void AddToLog(AnsiString S){
      if(LogMemo) LogMemo->Lines->Add("["+TimeToStr(Time())+"] "+S);
```

КОПІЯ 2025

```
    }
    void UpdateFrame(LPVIDEOHDR lpVHdr);
        void FilterFrame(BYTE* Src, BYTE* Dst);
        void PaintArray(BYTE * arr, Graphics::TBitmap * bmp);
        void CompareFrames(BYTE * f1, BYTE * f2, BYTE * dest);
        void DoAlarm();
        void SaveFrame();
        AnsiString EvalImgPath();

private:
        int inline CalcBufIndex(int arg);

} extern Sets;



class TMainForm : public TForm
{
__published:      // IDE-managed Components
        TMainMenu *MainMenu1;
        TMenuItem *SetItem;
        TMenuItem *DeviceItem;
        TPageControl *MainPageControl;
        TTabSheet *TabSheet1;
        TTabSheet *TabSheet2;
        TPanel *CaptPanel;
        TMemo *LogMemo;
        TTabSheet *TabSheet3;
        TPaintBox *FramePB;
        TMenuItem *FilterItem;
        TPopupMenu *TrayPopupMenu;
        TMenuItem *Show1;
        TTabSheet *TabSheet4;
        TPaintBox *FiltPB;
        TTabSheet *TabSheet5;
        TPaintBox *MotionPB;
        TPaintBox *PrevPB;
        TMenuItem *N1;
        TMenuItem *SaveItem;
        TMenuItem *CommonItem;
        TGroupBox *GroupBox1;
        TBitBtn *SetDeviceBitBtn;
        TBitBtn *SetFilterBitBtn;
        TBitBtn *SetCommonBitBtn;
        TStatusBar *StatusBar;
        TMenuItem *Hide1;
        TBitBtn *OpenImgFolderBitBtn;
        TActionList *ActionList1;
        TBrowseURL *BrowseURL1;
        TMenuItem *WebSite1;
        TGroupBox *GroupBox2;
        TBitBtn *ShootNowBitBtn;
        TButton *ActivateButton;
        TMenuItem *Language1;
        TMenuItem *EnglishItem;
        TMenuItem *RussianItem;
        void __fastcall ActivateButtonClick(TObject *Sender);
        void __fastcall DeviceItemClick(TObject *Sender);
        void __fastcall FormShow(TObject *Sender);
        void __fastcall FormClose(TObject *Sender, TCloseAction &Action);
        void __fastcall FilterItemClick(TObject *Sender);
        void __fastcall PrevPBPaint(TObject *Sender);
        void __fastcall SaveItemClick(TObject *Sender);
        void __fastcall LogMemoChange(TObject *Sender);
        void __fastcall Show1Click(TObject *Sender);
        void __fastcall Hide1Click(TObject *Sender);
        void __fastcall CommonItemClick(TObject *Sender);
        void __fastcall OpenImgFolderBitBtnClick(TObject *Sender);
        void __fastcall ShootNowBitBtnClick(TObject *Sender);
```

КСТЗ_2025

```
        void __fastcall FormCreate(TObject *Sender);
        void __fastcall EnglishItemClick(TObject *Sender);
private:    // User declarations
public:           // User declarations
        __fastcall TMainForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TMainForm *MainForm;
//---------------------------------------------------------------------------
#endif
```

КбПЗ_2025

// SetFilterUnit.cpp – модуль фільтрації відеозображення системи відеонагляду для IoT

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "SetFilterUnit.h"
#include "MainUnit.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TSetFilterForm *SetFilterForm;
int SRX1, SRX2, SRY1, SRY2;
int CurSens = -1;

//---------------------------------------------------------------------------
__fastcall TSetFilterForm::TSetFilterForm(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TSetFilterForm::OkBitBtnClick(TObject *Sender)
{
   DeleteFile("prev.cfg");
   Close();
}
//---------------------------------------------------------------------------
void __fastcall TSetFilterForm::CancelBitBtnClick(TObject *Sender)
{
   Sets.LoadFromFile("prev.cfg");
   DeleteFile("prev.cfg");
   Close();
}
//---------------------------------------------------------------------------
void __fastcall TSetFilterForm::SRPBPaint(TObject *Sender)
{
   DrawSR();
}
//---------------------------------------------------------------------------

void __fastcall TSetFilterForm::SRPBMouseMove(TObject *Sender,
      TShiftState Shift, int X, int Y)
{
  if(DrawRectButton->Down){
     SRPB->Cursor = crCross;
     if(Shift.Contains(ssLeft)){//натиснута ліва клавіша миші
        SRX2 = X;
        SRY2 = Y;
        DrawSR();
     }
  }
  else{
     SRPB->Cursor = crDefault;
  }
}
//---------------------------------------------------------------------------

void __fastcall TSetFilterForm::SRPBMouseDown(TObject *Sender,
      TMouseButton Button, TShiftState Shift, int X, int Y)
{
   if(Button == mbLeft && DrawRectButton->Down){
      SRX1 = SRX2 = X;
      SRY1 = SRY2 = Y;
   }

}
```

КБПЗ_2025

```
//---------------------------------------------------------------------------


void TSetFilterForm::DrawSR()
{
    TPicture *tPic = new TPicture;
    tPic->Bitmap->Width = SRPB->Width;
    tPic->Bitmap->Height =  SRPB->Height;
    tPic->Bitmap->Canvas->Brush->Color = (TColor)RGB(100,100,100);
    tPic->Bitmap->Canvas->FillRect(tPic->Bitmap->Canvas->ClipRect);
    tPic->Bitmap->Canvas->Brush->Color = clWhite;
    tPic->Bitmap->Canvas->FillRect(Rect(SRX1, SRY1, SRX2, SRY2));


    tPic->Bitmap->Canvas->CopyMode = cmSrcAnd;
    tPic->Bitmap->Canvas->StretchDraw(tPic->Bitmap->Canvas->ClipRect,
Sets.FramePic->Bitmap);
    SRPB->Canvas->Draw(0, 0, tPic->Bitmap);

    delete tPic;
}
void __fastcall TSetFilterForm::FormShow(TObject *Sender)
{
   //зберегти поточні налаштування
   Sets.SaveToFile("prev.cfg");

   SRPB->Width = Sets.pW;
   SRPB->Height = Sets.pH;
   SensPB->Width = Sets.pW;
   SensPB->Height = Sets.pH;
   if(Sets.ClipRect.bottom > 0){
      SRX1 = Sets.ClipRect.left;
      SRX2 = Sets.ClipRect.right;
      SRY1 = Sets.ClipRect.top;
      SRY2 = Sets.ClipRect.bottom;
   }else{
      SRX1 = SRY1 = 0;
      SRX2 = Sets.pW;
      SRY2 = Sets.pH;
   }

   ActivePointsEdit->Text = IntToStr(Sets.MaxActivePoints);
   AlarmTimeEdit->Text = IntToStr(Sets.AlarmMotionTime);
   CycleTimeEdit->Text = IntToStr(Sets.CycleTime);
   ForceSaveEdit->Text = IntToStr(Sets.ForceSaveTime);

}
//---------------------------------------------------------------------------

void __fastcall TSetFilterForm::RectSaveButtonClick(TObject *Sender)
{
   int t;
   if(SRX1 > SRX2){
      t = SRX1; SRX1 = SRX2; SRX2 = t;
   }

   if(SRY1 > SRY2){
      t = SRY1; SRY1 = SRY2; SRY2 = t;
   }

   Sets.ClipRect.left = SRX1;
   Sets.ClipRect.right = SRX2;
   Sets.ClipRect.top = SRY1;
   Sets.ClipRect.bottom = SRY2;
   if(Sets.bConnected) capGrabFrame(Sets.hCaptureW);
}
//---------------------------------------------------------------------------

void __fastcall TSetFilterForm::SensPBPaint(TObject *Sender)
```

КБПЗ_2025

```
{
   DrawSensRegions();
}
//---------------------------------------------------------------------------


void TSetFilterForm::DrawSensRegions()
{
   int i;
   SensRegion *sR;
   TPicture *tPic = new TPicture;
   tPic->Bitmap->Width = SensPB->Width;
   tPic->Bitmap->Height =  SensPB->Height;

   for(i = 0; i < Sets.SensList->Count; i++){
      sR = (SensRegion*)Sets.SensList->Items[i];
      int cl = sR->sens * 255.0 / 100.0;
      tPic->Bitmap->Canvas->Brush->Color = (TColor)RGB(cl, cl, cl);
      tPic->Bitmap->Canvas->FillRect(*sR);
   }

   tPic->Bitmap->Canvas->CopyMode = cmSrcAnd;
   tPic->Bitmap->Canvas->Draw(0, 0, Sets.GrayPic->Bitmap);

   tPic->Bitmap->Canvas->Font->Color = clRed;
   tPic->Bitmap->Canvas->Pen->Width = 1;

   for(i = 0; i < Sets.SensList->Count; i++){
      sR = (SensRegion*)Sets.SensList->Items[i];
      tPic->Bitmap->Canvas->Brush->Style = bsSolid;
      tPic->Bitmap->Canvas->Brush->Color = clRed;
      tPic->Bitmap->Canvas->FrameRect(*sR);
      tPic->Bitmap->Canvas->Brush->Style = bsClear;
      tPic->Bitmap->Canvas->TextOut((sR->left+sR->right)*0.5-10, (sR->top+sR-
>bottom)*0.5-10, IntToStr(sR->sens));
   }

   SensPB->Canvas->Draw(0, 0, tPic->Bitmap);

   delete tPic;
  // SensPB->Canvas->Draw(0, 0, Sets.GrayPic->Bitmap);

}

void __fastcall TSetFilterForm::SensSpeedButtonClick(TObject *Sender)
{
   if(!((TSpeedButton*)Sender)->Down){
      CurSens = -1;
   }else{
      CurSens = StrToInt(((TSpeedButton*)Sender)->Caption);
   }

}
//---------------------------------------------------------------------------


void __fastcall TSetFilterForm::SensPBMouseUp(TObject *Sender,
      TMouseButton Button, TShiftState Shift, int X, int Y)
{
   int i;
   SensRegion *sR;
   if(CurSens != -1 && Button == mbLeft){
      for(i = 0; i < Sets.SensList->Count; i++){
         sR = (SensRegion*)Sets.SensList->Items[i];
         if(sR->left < X && sR->right > X && sR->top < Y && sR->bottom > Y){
            sR->sens = CurSens;
         }
      }
      DrawSensRegions();
```

КОПЗ_2025

```
   }
}
//---------------------------------------------------------------------------

void __fastcall TSetFilterForm::SensPBMouseMove(TObject *Sender,
      TShiftState Shift, int X, int Y)
{
   if(CurSens != -1){
      SensPB->Cursor = crHandPoint;
   }else{
      SensPB->Cursor = crDefault;
   }
}
//---------------------------------------------------------------------------

void __fastcall TSetFilterForm::SensClearButtonClick(TObject *Sender)
{
   int i;
   SensRegion *sR;
   for(i = 0; i < Sets.SensList->Count; i++){
      sR = (SensRegion*)Sets.SensList->Items[i];
      sR->sens = 90;
   }
   DrawSensRegions();
}
//---------------------------------------------------------------------------



void __fastcall TSetFilterForm::AlarmApplyBitBtnClick(TObject *Sender)
{
   try{
      Sets.MaxActivePoints = StrToInt(ActivePointsEdit->Text.Trim());
      Sets.AlarmMotionTime = StrToInt(AlarmTimeEdit->Text.Trim());
      Sets.CycleTime = StrToInt(CycleTimeEdit->Text.Trim());
      Sets.ForceSaveTime = StrToInt(ForceSaveEdit->Text.Trim());
   }catch(...){
      ShowMessage("Введене число не входить в допустимий діапазон значень!");
   }
}
//---------------------------------------------------------------------------
```

КОПЗ_2025

// SetFilterUnit.h – заголовочний файл модуля фільтрації відеозображення системи відеонагляду для IoT

```
//---------------------------------------------------------------------------

#ifndef SetFilterUnitH
#define SetFilterUnitH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <Buttons.hpp>
#include <ComCtrls.hpp>
#include <ExtCtrls.hpp>
//---------------------------------------------------------------------------
class TSetFilterForm : public TForm
{
__published:      // IDE-компоненти
        TBitBtn *OkBitBtn;
        TBitBtn *CancelBitBtn;
        TPageControl *PageControl1;
        TTabSheet *TabSheet1;
        TPaintBox *SRPB;
        TTabSheet *TabSheet2;
        TBitBtn *RectSaveButton;
        TSpeedButton *DrawRectButton;
        TPaintBox *SensPB;
        TBitBtn *SensClearButton;
        TSpeedButton *SensSpeedButton;
        TSpeedButton *SpeedButton2;
        TSpeedButton *SpeedButton3;
        TSpeedButton *SpeedButton4;
        TSpeedButton *SpeedButton5;
        TSpeedButton *SpeedButton6;
        TSpeedButton *SpeedButton7;
        TSpeedButton *SpeedButton8;
        TSpeedButton *SpeedButton9;
        TSpeedButton *SpeedButton10;
        TSpeedButton *SpeedButton11;
        TTabSheet *TabSheet3;
        TLabel *Label1;
        TLabel *Label2;
        TLabel *Label3;
        TLabel *Label4;
        TLabel *Label5;
        TLabel *Label6;
        TLabel *Label7;
        TEdit *CycleTimeEdit;
        TEdit *ActivePointsEdit;
        TEdit *AlarmTimeEdit;
        TEdit *ForceSaveEdit;
        TLabel *Label8;
        TBitBtn *AlarmApplyBitBtn;
        void __fastcall OkBitBtnClick(TObject *Sender);
        void __fastcall CancelBitBtnClick(TObject *Sender);
        void __fastcall SRPBPaint(TObject *Sender);
        void __fastcall SRPBMouseMove(TObject *Sender, TShiftState Shift,
          int X, int Y);
        void __fastcall SRPBMouseDown(TObject *Sender, TMouseButton Button,
          TShiftState Shift, int X, int Y);
        void __fastcall FormShow(TObject *Sender);
        void __fastcall RectSaveButtonClick(TObject *Sender);
        void __fastcall SensPBPaint(TObject *Sender);
        void __fastcall SensSpeedButtonClick(TObject *Sender);
        void __fastcall SensPBMouseUp(TObject *Sender, TMouseButton Button,
          TShiftState Shift, int X, int Y);
        void __fastcall SensPBMouseMove(TObject *Sender, TShiftState Shift,
          int X, int Y);
```

КОТБ_2025

```
        void __fastcall SensClearButtonClick(TObject *Sender);
        void __fastcall AlarmApplyBitBtnClick(TObject *Sender);
private:    // користувацькі об'яви
public:           // користувацькі об'яви
        __fastcall TSetFilterForm(TComponent* Owner);
        void DrawSR();
        void DrawSensRegions();
};
//---------------------------------------------------------------------------
extern PACKAGE TSetFilterForm *SetFilterForm;
//---------------------------------------------------------------------------
#endif
```

КбПЗ_2025

// SetDeviceUnit.cpp – модуль вибору пристрою для ведення відеоспостереження

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "SetDeviceUnit.h"
#include "MainUnit.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TSetDeviceForm *SetDeviceForm;
//---------------------------------------------------------------------------
__fastcall TSetDeviceForm::TSetDeviceForm(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TSetDeviceForm::FormShow(TObject *Sender)
{
  //зберегти поточні налаштування
  Sets.SaveToFile("prev.cfg");

  UpdateState();

  // перерахування пристроїв
  char szDeviceName[80];
  char szDeviceVersion[80];

  DeviceSelect->Items->Clear();
  for (int wIndex = 0; wIndex < 10; wIndex++)
  {
    if (capGetDriverDescription (wIndex, szDeviceName,
        sizeof (szDeviceName), szDeviceVersion,
        sizeof (szDeviceVersion)))
    {
        // Додати ім’я до списку встановлених драйверів захвату
        // та потім дозволити користувачеві вибрати драйвер для використання
        DeviceSelect->Items-
>Add(AnsiString(szDeviceName)+"("+AnsiString(szDeviceVersion)+")");
        if(Sets.DevIndex == wIndex) DeviceSelect->Text =
AnsiString(szDeviceName)+"("+AnsiString(szDeviceVersion)+")";
    }
  }
}
//---------------------------------------------------------------------------
void __fastcall TSetDeviceForm::DeviceSelectSelect(TObject *Sender)
{
   //update device index
   Sets.DevIndex = DeviceSelect->ItemIndex;
   UpdateState();


}
//---------------------------------------------------------------------------
void __fastcall TSetDeviceForm::OkBitBtnClick(TObject *Sender)
{
  DeleteFile("prev.cfg");
  Close();
}
//---------------------------------------------------------------------------
void __fastcall TSetDeviceForm::CancelBitBtnClick(TObject *Sender)
{
  Sets.LoadFromFile("prev.cfg");
  DeleteFile("prev.cfg");
  Close();
}
//---------------------------------------------------------------------------
```

КОПІЗ_2025

```
void __fastcall TSetDeviceForm::VideoSrcBitBtnClick(TObject *Sender)
{
   capDlgVideoSource(Sets.hCaptureW);
}
//---------------------------------------------------------------------------
void __fastcall TSetDeviceForm::FormatBitBtnClick(TObject *Sender)
{
   capDlgVideoFormat(Sets.hCaptureW);
   capGetStatus(Sets.hCaptureW, &Sets.CapStatus, sizeof (CAPSTATUS));

   Sets.VideoFormatSize = capGetVideoFormatSize(Sets.hCaptureW);
   if(Sets.VideoFormat){
      delete[] Sets.VideoFormat;
      Sets.VideoFormat = NULL;
   }
   if(Sets.VideoFormatSize > 0){
      Sets.VideoFormat = (LPBITMAPINFO) new BYTE[Sets.VideoFormatSize];
      capGetVideoFormat(Sets.hCaptureW, Sets.VideoFormat, Sets.VideoFormatSize);
   }
}
//---------------------------------------------------------------------------
void __fastcall TSetDeviceForm::DisplayBitBtnClick(TObject *Sender)
{
   capDlgVideoDisplay(Sets.hCaptureW);
}
//---------------------------------------------------------------------------


void TSetDeviceForm::UpdateState()
{
   //під'єднати новий пристрій
   capDriverDisconnect (Sets.hCaptureW);
   bool fOK = capDriverConnect(Sets.hCaptureW, Sets.DevIndex);

   if(fOK){
      //отримати заголовки драйверу
      capDriverGetCaps(Sets.hCaptureW, &Sets.CapDrvCaps, sizeof
(CAPDRIVERCAPS));
      //отримати заголовки вікна
      capGetStatus(Sets.hCaptureW, &Sets.CapStatus, sizeof (CAPSTATUS));
      //отримати доступні діалоги для драйверу
      // діалогове вікно для джерела відеосигналу
      VideoSrcBitBtn->Visible = false;
      if (Sets.CapDrvCaps.fHasDlgVideoSource)
         VideoSrcBitBtn->Visible = true;
      // діалогове вікно для формату відео
      FormatBitBtn->Visible = false;
      if (Sets.CapDrvCaps.fHasDlgVideoFormat)
         FormatBitBtn->Visible = true;
      // діалогове вікно для відеозображення системи відеонагляду для IoT
      DisplayBitBtn->Visible = false;
      if (Sets.CapDrvCaps.fHasDlgVideoDisplay)
         DisplayBitBtn->Visible = true;

   }
}
```

КСПЗ-2025

// SetDeviceUnit.h – заголовочний файл модуля вибору пристрою для ведення відеоспостереження

```
//---------------------------------------------------------------------------

#ifndef SetDeviceUnitH
#define SetDeviceUnitH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <Buttons.hpp>
//---------------------------------------------------------------------------
class TSetDeviceForm : public TForm
{
__published:      // IDE-managed Components
        TComboBox *DeviceSelect;
        TBitBtn *OkBitBtn;
        TBitBtn *CancelBitBtn;
        TBitBtn *VideoSrcBitBtn;
        TBitBtn *FormatBitBtn;
        TBitBtn *DisplayBitBtn;
        TLabel *Label1;
        void __fastcall FormShow(TObject *Sender);
        void __fastcall DeviceSelectSelect(TObject *Sender);
        void __fastcall OkBitBtnClick(TObject *Sender);
        void __fastcall CancelBitBtnClick(TObject *Sender);
        void __fastcall VideoSrcBitBtnClick(TObject *Sender);
        void __fastcall FormatBitBtnClick(TObject *Sender);
        void __fastcall DisplayBitBtnClick(TObject *Sender);
private:    // користувацькі об'яви
public:           // користувацькі об'яви
        __fastcall TSetDeviceForm(TComponent* Owner);
        void UpdateState();
};
//---------------------------------------------------------------------------
extern PACKAGE TSetDeviceForm *SetDeviceForm;
//---------------------------------------------------------------------------
#endif
```

КОПЗ_2025

// SetCommonUnit.cpp – модуль вибору параметрів відеоспостереження

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "SetCommonUnit.h"
#include "MainUnit.h"

//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma link "cdiroutl"
#pragma link "CSPIN"
#pragma resource "*.dfm"
TSetCommonForm *SetCommonForm;
//---------------------------------------------------------------------------
__fastcall TSetCommonForm::TSetCommonForm(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TSetCommonForm::OkBitBtnClick(TObject *Sender)
{
   Sets.ImagesFolder = ImageDirEdit->Text;
   Sets.JPEGCompression = JPEGCompressionEdit->Value;
   Sets.AlarmExecuteFN = AlarmOpenEdit->Text;
   DeleteFile("prev.cfg");
   Close();
}
//---------------------------------------------------------------------------
void __fastcall TSetCommonForm::CancelBitBtnClick(TObject *Sender)
{
   Sets.LoadFromFile("prev.cfg");
   DeleteFile("prev.cfg");
   Close();
}
//---------------------------------------------------------------------------
void __fastcall TSetCommonForm::FormShow(TObject *Sender)
{
   //зберегти поточні налаштування
   Sets.SaveToFile("prev.cfg");

   DLB->Directory = Sets.ImagesFolder;
   ImageDirEdit->Text = Sets.ImagesFolder;
   FNTemplateEdit->Text = Sets.FNTemplate;
   FNExampleLabel->Caption = Sets.EvalImgPath();
   JPEGCompressionEdit->Value = Sets.JPEGCompression;
   AlarmOpenEdit->Text = Sets.AlarmExecuteFN;
   AlarmOpenDialog->FileName = Sets.AlarmExecuteFN;
}
//---------------------------------------------------------------------------
void __fastcall TSetCommonForm::ImagePathSpeedButtonClick(TObject *Sender)
{
   DLPanel->Show();
}
//---------------------------------------------------------------------------
void __fastcall TSetCommonForm::DLOkButtonClick(TObject *Sender)
{
  ImageDirEdit->Text = DLB->Directory;
  DLPanel->Hide();
  Sets.ImagesFolder = ImageDirEdit->Text;
  Sets.FNTemplate = FNTemplateEdit->Text;
  FNExampleLabel->Caption = Sets.EvalImgPath();
}
//---------------------------------------------------------------------------
void __fastcall TSetCommonForm::DLCancelButtonClick(TObject *Sender)
{
```

КПЗ_2025

```
   DLPanel->Hide();
}
//---------------------------------------------------------------------------
void __fastcall TSetCommonForm::AlarmExecuteSpeedButtonClick(
      TObject *Sender)
{
   if(AlarmOpenDialog->Execute()){
      AlarmOpenEdit->Text = AlarmOpenDialog->FileName;
   }
}
//---------------------------------------------------------------------------
void __fastcall TSetCommonForm::OnAlarmNOExecuteSpeedButtonClick(
      TObject *Sender)
{
   AlarmOpenEdit->Clear();
}
//---------------------------------------------------------------------------

void __fastcall TSetCommonForm::FNTemplateEditKeyUp(TObject *Sender,
      WORD &Key, TShiftState Shift)
{
   Sets.FNTemplate = FNTemplateEdit->Text;
   FNExampleLabel->Caption = Sets.EvalImgPath();
}
//---------------------------------------------------------------------------


void __fastcall TSetCommonForm::HideFNHelpButtonClick(TObject *Sender)
{
  FNHelpBox->Visible = false;
}
//---------------------------------------------------------------------------

void __fastcall TSetCommonForm::SpeedButton2Click(TObject *Sender)
{
  FNHelpBox->Visible = true;
}
//---------------------------------------------------------------------------

void __fastcall TSetCommonForm::ClearFNTemplateButtonClick(TObject *Sender)
{
  FNTemplateEdit->Text = "{dd}_{mm}_{yyyy}\\{dd}{mmm}{yy}-{hh}{nn}{ss}{zzz}";
  Sets.FNTemplate = FNTemplateEdit->Text;
  FNExampleLabel->Caption = Sets.EvalImgPath();
}
//---------------------------------------------------------------------------
```

КОПИЯ 2025

// SetCommonUnit.h – заголовочний файл модуля вибору параметрів відеоспостереження

```
//---------------------------------------------------------------------------

#ifndef SetCommonUnitH
#define SetCommonUnitH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <Buttons.hpp>
#include "cdiroutl.h"
#include <Grids.hpp>
#include <Outline.hpp>
#include <FileCtrl.hpp>
#include <ExtCtrls.hpp>
#include "CSPIN.h"
#include <Dialogs.hpp>


//---------------------------------------------------------------------------


class TSetCommonForm : public TForm
{
__published:      // IDE-managed Components
        TBitBtn *OkBitBtn;
        TBitBtn *CancelBitBtn;
        TEdit *ImageDirEdit;
        TSpeedButton *ImagePathSpeedButton;
        TPanel *DLPanel;
        TDirectoryListBox *DLB;
        TDriveComboBox *DCB;
        TButton *DLOkButton;
        TButton *DLCancelButton;
        TLabel *Label1;
        TCSpinEdit *JPEGCompressionEdit;
        TLabel *Label2;
        TLabel *Label3;
        TEdit *AlarmOpenEdit;
        TSpeedButton *AlarmExecuteSpeedButton;
        TOpenDialog *AlarmOpenDialog;
        TSpeedButton *OnAlarmNOExecuteSpeedButton;
        TGroupBox *GroupBox1;
        TMemo *AboutMemo;
        TLabel *Label4;
        TEdit *FNTemplateEdit;
        TLabel *FNExampleLabel;
        TSpeedButton *ClearFNTemplateButton;
        TSpeedButton *SpeedButton2;
        TGroupBox *FNHelpBox;
        TMemo *Memo1;
        TButton *HideFNHelpButton;
        void __fastcall OkBitBtnClick(TObject *Sender);
        void __fastcall CancelBitBtnClick(TObject *Sender);
        void __fastcall FormShow(TObject *Sender);
        void __fastcall ImagePathSpeedButtonClick(TObject *Sender);
        void __fastcall DLOkButtonClick(TObject *Sender);
        void __fastcall DLCancelButtonClick(TObject *Sender);
        void __fastcall AlarmExecuteSpeedButtonClick(TObject *Sender);
        void __fastcall OnAlarmNOExecuteSpeedButtonClick(TObject *Sender);
        void __fastcall FNTemplateEditKeyUp(TObject *Sender, WORD &Key,
          TShiftState Shift);
        void __fastcall HideFNHelpButtonClick(TObject *Sender);
        void __fastcall SpeedButton2Click(TObject *Sender);
        void __fastcall ClearFNTemplateButtonClick(TObject *Sender);
```

КПЗ_2025

```
private:    // користувацькі об’яви
public:           // користувацькі об’яви
        __fastcall TSetCommonForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TSetCommonForm *SetCommonForm;
//---------------------------------------------------------------------------
#endif
```

КбПЗ_2025