МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Кафедра комп'ютеризованих систем управління

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри

_____Литвиненко О.Є.

«___» _____2021 р.

ДИПЛОМНИЙ ПРОЄКТ

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ
"БАКАЛАВР"

Тема: Апаратно-програмний комплекс моніторингу доступу до приміщень

Виконавець: Дусь О.В.

Керівник: Росінська Г.П.

Нормоконтролер: Тупота Є.В.

Київ 2020

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, комп'ютерної та програмної інженерії
Кафедра комп'ютеризованих систем управління
Спеціальність 123 "Комп'ютерна інженерія"
(шифр, найменування)

Освітньо професійна програма «Системне програмування»
Форма навчання заочна

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____Литвиненко О. Є.

«_____»_____2020 р.

ЗАВДАННЯ
на виконання дипломного проєкту

Дуся Олега Вікторовича
(прізвище, ім'я, по батькові)

1.Тема роботи: "Апаратно-програмний комплекс моніторингу доступу до приміщень "
затверджена наказом ректора від "21" грудня 2020 року № 2523 /ст.

2. Термін виконання роботи: з 11.01.2021 до 28.02.2021

3. Вихідні дані до роботи: 1) вимоги до програми;
2) основні операції контролю доступу до приміщень.

4.Зміст пояснювальної записки (перелік питань, що підлягають розробці):
1) аналіз принципів роботи систем контролю і управління доступом;
2) принципи роботи систем відеоспостереження;
3) описання програмної реалізації інтеграції системи контролю управління доступом з системою відеоспостереження.

5.Перелік обов'язкового графічного матеріалу:
1) діаграма класів механізму складування об'єктів;
2) діаграма класів механізму команд;
3) основні модулі пакету «Система зв'язку сервера»;
4) схема алгоритму занесення даних з камер до файлів сервера.

6. Календарний план-графік

№ п/п	Етапи виконання дипломного проєкту	Термін виконання етапів	Примітка
1	Провести аналіз літератури за темою дипломного проєкту та аналіз існуючих систем	11.01.21 12.01.21	
2	Зробити вибір компонентів системи	13.01.21- 14.01.21	
3	Розробити структуру програмних засобів системи цифрового репозитарію	15.01.21- 17.01.21	
4	Розробити програмні засоби цифрового репозитарію	18.01.21- 29.01.21	
5	Провести налаштування програмних засобів на сервері	30.01.21- 02.02.21	
6	Написати пояснювальну записку	03.02.21- 14.02.21	
7	Підготувати презентацію	15.02.21- 21.02.21	

7. Дата видачі завдання «_11_» січня 2021 р.

Керівник дипломного проекту_____Росінська Г.П.
(підпис)

Завдання прийняв до виконання_____Дусь О.В.
(підпис студента)

РЕФЕРАТ

Пояснювальна записка до дипломного проєкту “Апаратно-програмний комплекс моніторингу доступу до приміщень”: 63 с., 16 рис., 19 літературних джерел, 1 додаток.

ІНТЕГРОВАНІ ОХОРОННІ СИСТЕМИ, НЕСАНКЦІОНОВАНИЙ ДОСТУП, СИСТЕМА КОМПЛЕКСНОЇ БЕЗПЕКИ, КЛАСИ ЗАХИСТУ, ТЕХНІЧНЕ ОБСЛУГОВУВАННЯ ОХОРОНИ ПРИМІЩЕНЬ, ПРОГРАМНА ІНТЕГРАЦІЯ КОМПОНЕНТІВ ОХОРОННОЇ СИСТЕМИ

Об’єкт дослідження – інтеграція системи контролю доступу з системою відеоспостереження для моніторингу доступу до приміщень.

Предмет дослідження – апаратно-програмний комплекс моніторингу доступу до приміщень.

Метою роботи є інтеграція системи контролю доступу з системою відеоспостереження.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ, ТЕРМІНІВ

АС	– автоматизована система
БД	– база даних
ЕОМ	– електронно-обчислювальна машина
ЗВ	– засоби відображення
ОС	– операційна система
ОЗП	– оперативний запам’ятовуючий пристрій
ППП	– пакет прикладних програм
СКУД	– система керування управлінням доступом

ВСТУП

В наш час будь-яка організація, що має в своєму розпорядженні конфіденційну чи секретну інформацію, зобов'язана мати систему контролю доступу.

СКУД зарекомендували себе як надійні, гнучкі і функціональні. У переліку вироблюваних різними компаніями продуктів для системи контролю управління доступом містяться контролери з централізованою архітектурою, аналогові і цифрові панелі охоронної сигналізації, релейні модулі, зчитувачі і проміжні блоки з власною пам'яттю і вбудованою логікою, здатні працювати автономно, цифрові і аналогові інтерфейси управління кінцевими пристроями і ін. З допомогою контролера система може управляти різними виконавчими пристроями. Звичайно це електромеханічний замок, турнікет, автоматичні ворота та ін.

Системи управління і контролю доступом в даний час є невід'ємною частиною багатьох сучасних систем безпеки. Центром системи контролю доступу є устаткування, що управляє, – контролери. Але системи контролю доступу призначені ще і для відстежування пересувань. Наприклад, можна відстежувати переміщення співробітників усередині будівлі для автоматичного складання табеля обліку робочого часу. Устаткування систем контролю і управління доступом ділиться на автономні і мережеві системи. Автономні системи контролю доступу мають зчитувачі того або іншого типу, або клавіатуру для набору коду і контролер, що пам'ятає легальні коди та керує замком або іншим виконавчим механізмом.

Мережеві системи контролю доступу включають декілька зчитувачів і один або декілька контролерів. З використанням додаткових інтерфейсних модулів можливе підключення будь-яких зчитувачів.

Сучасні системи контролю доступу мають безліч застосувань, а у кожного замовника – індивідуальні запити. Крім вирішення питань контролю доступу, система здійснює облік робочого часу співробітників, що приводить до

підвищення трудової дисципліни і мотивації персоналу. Існують також автономні системи контролю доступу з накопиченням інформації про всі переміщення через точку контролю (двері, шлагбаум, турнікет) – час, дата, ідентифікаційний номер *Proximity*- або *smart*-карти або брелока *Touch Memory*. Вся інформація зберігається в пам'яті контролера. Системи контролю управління доступом дозволяють автоматично контролювати вхід людей в будівлю або приміщення і вихід з нього, а також в'їзд автотранспорту на територію і виїзд. Таким чином, установка СКУД просто необхідна тим, хто приділяє належну увагу безпеці. Мережеві системи поєднують в собі функції контролю і управління доступом і охоронної сигналізації, що дозволяє забезпечити комплексний захист об'єкту без використання додаткових засобів.

Недорогі і зручні рішення, коли потрібно управляти доступом в невеликі офіси. Контролери, що застосовуються в монтажі мережевих систем контролю доступу, підключаються до комп'ютера і передають інформацію про проходи через контрольовані двері або турнікет.

СКУД дозволить запобігти доступу небажаних осіб, а співробітникам точно вказати ті приміщення, в які вони мають право доступу. Складніша система дозволить, крім обмеження доступу, призначити кожному співробітникові індивідуальний часовий графік роботи, зберегти і потім проглянути інформацію про події за день. Системи можуть працювати в автономному режимі і під управлінням комп'ютера. СКУД дозволить створити системи контролю доступу будь-якої складності з можливістю контролю і управління проходу співробітників в різні приміщення.

Особливістю можна виділити те, що, ухваливши рішення про придбання, клієнт часто не представляє, скільки можливостей може дати ця система. Сучасні технології дозволяють не просто контролювати безпосередньо доступ на об'єкт, але і створювати різні рівні доступу.

СКУД – система контролю і управління доступом – знаходить широке застосування у сфері забезпечення безпеки. Без системи контролю доступу не обходиться жоден солідний офіс. Мережеві системи призначені для забезпечення контролю і управління доступом на крупних об'єктах (банки,

установи, підприємства та ін.). Контролери забезпечують контроль і управління доступом, в одне або декілька приміщень, об'єднуються в мережу (як правило на основі інтерфейсу *RS*-485) і підключаються до комп'ютера, що управляє.

Рано чи пізно, на будь-якому сучасному підприємстві виникає проблема обмеження доступу сторонніх осіб на територію підприємства або в приміщення. У будь-якому офісі є місця, куди не повинні заходити звичайні відвідувачі (бухгалтерія, склад та ін.) І будь-який керівник прагне контролювати дисципліну співробітників і їх доступ в кабінети і приміщення підприємства або офісу

При правильному підході до проектування і раціональному виборі обладнання кожна підсистема успішно вирішує ті завдання, для яких вона призначена. Але розрізнені системи не забезпечують прийнятного рівня безпеки об'єкта.

Тенденції сучасного розвитку систем безпеки нерозривно пов'язані з процесами широкої автоматизації та інтеграції. Логічним розвитком такої інтеграції стало створення інтегрованих систем безпеки (ІСБ) з широкими функціональними можливостями, що дозволяють автоматизувати також управління інженерними системами будівлі або об'єкта. Основою таких ІСБ служить єдина апаратно-програмна платформа, що представляє собою автоматизовану систему управління (АСУ) з багаторівневою мережевою структурою.

Основною ознакою інтегрованої системи безпеки (ІСБ) є спільне використання ресурсів підсистем, в результаті чого система як ціле набуває якісно нових властивостей, на відміну від випадку автономної роботи підсистем. Актуальність теми випускної кваліфікаційної роботи полягає в тому, що інтеграція – якісно новий етап в побудові систем безпеки.

Необхідність інтеграції сьогодні обумовлюється різноманітними причинами, на перше місце з яких виходить розширення сфери автоматизації діяльності організацій. Так, наприклад, якщо раніше під автоматизацією в сфері безпеки розумілися такі базові можливості, як автоматичне пожежогасіння, управління доступом на об'єкт, то сьогодні актуальними є вимоги об'єднання

систем безпеки з системами життєзабезпечення, технологічними підсистемами, управлінням виробництвом, документообігом, бухгалтерією і т.д.

Об’єктом дослідження є інтеграція системи контролю доступу з системою відеоспостереження. Предметом – модифікована система контролю доступу.

Метою роботи є інтеграція системи контролю доступу з системою відеоспостереження.

Відповідно до даної метою необхідно вирішити наступні завдання:

– розглянути види систем контролю і управління доступом;

– розглянути систему контролю і управління доступом як базовий компонент інтегрованих систем;

– вивчити технічні тенденції розвитку систем контролю і управління доступом;

– розглянути структуру і основні елементи систем відеоспостереження;

– визначити області застосування і програмне забезпечення систем відеоспостереження;

– розглянути інтегровані системи безпеки.

РОЗДІЛ 1
АНАЛІЗ ПРИНЦИПІВ РОБОТИ СИСТЕМ КОНТРОЛЮ І УПРАВЛІННЯ ДОСТУПОМ

1.1. Проблеми інформаційної безпеки підприємств та організацій

Проблема інформаційної безпеки підприємств та організацій є надзвичайно актуальною на сучасному етапі розвитку інформаційних технологій, інформаційних систем і мереж. Це пояснюється зростаючими технічними і програмними можливостями доступу до інформації, що не завжди є правомірним. Актуальність проблеми інформаційної безпеки підприємств і організацій визначається рядом взаємозв'язаних факторів, більшість з яких є наслідком процесу інформатизації сучасного суспільства. Серед таких факторів, з одного боку, – формування правових засад інформатизації, поширення застосування сучасних інформаційних технологій у підприємницькій діяльності, а з іншого – висока уразливість інформаційних систем, стрімкий прогрес розвитку так званої «інформаційної зброї». Особливості соціально-економічної ситуації, відсутність реальних обмежень щодо доступу до засобів інформаційного нападу призводять до численних фактів їх застосування конкурентами, кримінальними елементами, іншими суб'єктами проти комерційних структур.

Крім того, в умовах жорстокої конкурентної боротьби, проведення суб'єктами економічних ринків актів недобросовісної конкуренції та дій, пов'язаних із промисловим шпигунством, виникла нагальна потреба в розмежуванні доступу до носіїв інформації безпосередньо на підприємствах,

виокремлення інформації з обмеженим доступом і визначення порядку її отримання та використання.

Підприємницька діяльність досить насичена не тільки діловими відносинами, їй значною мірою притаманні тісні інформаційні стосунки партнерів та конкурентів. Останні ж регулюються відповідними законодавчими та нормативними актами держави, а також нормативними документами підприємницьких структур

Згідно зі ст. 60 Закону України «Про банки і банківську діяльність» до банківської таємниці належить інформація про діяльність і фінансовий стан клієнта, що стала відома банку у процесі його обслуговування і взаємовідносин із ним або з третіми особами під час надання послуг банком, розголошення якої може завдати матеріальної чи моральної шкоди.

Тлумачення поняття комерційної таємниці дається у ст. 30 Закону України «Про підприємства в Україні». Зокрема у статті вказується, що під комерційною таємницею підприємства розуміють відомості, пов'язані з виробництвом, технологічною інформацією, управлінням фінансами та іншою діяльністю підприємства, що не є державною таємницею, розголошення (передання, витік) яких може завдати шкоди його інтересам.

Відповідно до ч. 3 ст. 30 Закону України «Про інформацію» власникам конфіденційної інформації надано право самим включати її до категорії конфіденційної, визначати режим доступу до неї та встановлювати систему (способи) її захисту.

Комерційна таємниця підприємства і проблеми її охорони і захисту розглядаються, як комплексна задача. Під комерційною цінністю інформації можна розуміти грошовий еквівалент, який в даному випадку може бути сплачений за право володіння відомостями. Конкуренти і інші суб'єкти, прямо або побічно пов'язані з підприємством, об'єктивно зацікавлені в зборі і отриманні інформації самого різного роду. Потенційна комерційна цінність відомостей має місце у разі, коли при певних обставинах, в певний час і у визначеному місці з'являється або може з'явитися інтерес третіх осіб до придбання інформації про підприємство [3].

Вільний доступ до відомостей означає відсутність заходів, направлених на охорону і захист інформації від спроб ознайомлення з нею третіх осіб. У вільному доступі, як правило, знаходяться загальні довідкові відомості про підприємство – реквізити, історія створення, основні напрями діяльності підприємства в тій частині, в якій вони не захищені положенням про комерційну таємницю. Деякі підприємства, які проводять політику інформаційної відвертості і організаційної прозорості для необмеженого кола осіб, можуть залежно від специфіки своєї діяльності розміщувати інформацію про своїх працівників, ціни на продукцію, що випускається, або послуги, що надаються. У ряді випадків у вільному доступі знаходяться повідомлення прес-служби або іншого підрозділу, адресовані засобам масової інформації, які можуть стосуватися різних питань діяльності підприємства.

Охорона конфіденційності інформації підприємством може здійснюватися, в першу чергу, на підставі виданих локальних нормативних актів, найважливішим з яких є «Положення про комерційну таємницю». Всі заходи безпеки, захисту інформації від несанкціонованого доступу ззовні і усередині підприємства з боку не уповноважених співробітників повинні базуватися саме на положеннях даного документа

Окремі відомості, які доцільно відносити до предмету комерційної таємниці по сферах і характеру діяльності підприємства можна розділити на наступні групи відомостей:

– відомості про фінансову діяльність;

– інформація про ринок;

– відомості про виробництво, виконання робіт і надання послуг;

– відомості про наукові розробки;

– відомості про систему матеріально-технічного забезпечення;

– відомості про персонал підприємства;

– відомості про принципи управління підприємством;

– інші відомості.

Параметри діяльності підприємства, виражені в грошовому еквіваленті, можуть дати обширне і досить вичерпне уявлення про його фінансовий стан,

наявність заборгованості, об’єм оборотних коштів, географія вироблюваних платежів і так далі.

Інформація про використовувані підприємством кредити може бути використана для визначення і з’ясування особливостей нового напряму діяльності (комерційних задумів і проектів) [11].

Вироблювані підприємством маркетингові дослідження, а також експерименти, пов’язані із залученням потенційних клієнтів до вироблюваної продукції або послуг, що надаються, представляють інтерес як готовий інформаційний продукт, отриманий в ході витратної діяльності. Інформація такого роду повинна охоронятися особливо ретельно, оскільки саме з її допомогою підприємство може добитися істотного збільшення рентабельності своєї діяльності, збільшити свою частку на товарних ринках або ринках послуг.

Відомості про виробництво, виконання робіт і надання послуг розглядають об’єм і асортимент продукції, що випускається, або специфіка що надаються є однією з найважливіших економічних характеристик підприємства. Дані характеристики можуть також дати деякі уявлення про використовувані підприємством технології, про наявність або відсутність власного дослідницького підрозділу і тому подібне.

Відомості про наукові розробки – це перспективні високотехнологічні галузі української економіки, такі як розробка програмного забезпечення, телекомунікація, біотехнологія, виробництво устаткування, літакобудування, космонавтика і тому подібне. Наукові розробки можуть бути як на великих, так і на середніх і малих підприємствах, тому особливо важливо не нехтувати заходами безпеки в невеликих колективах – наявність інтересу ззовні залежить не від розміру підприємства, а від наявності перспективних розробок.

Відомості про систему матеріально-технічного забезпечення передбачають зведення про склад клієнтів, представників і посередників, потребу в матеріалах, сировині, аксесуарах і деталях конструкцій і інтер’єрів, джерела задоволення цих потреб, транспортні і енергетичні потреби підприємства.

Відомості про персонал підприємства надають кількісні і якісні характеристики персоналу дають третім особам можливість за допомогою

аналітичних дій робити висновки про деякі інші його параметри – про продуктивність праці і, таким чином, рівень виробництва, про наявність контактів з іноземними постачальниками або клієнтами, що набувають продукції або послугами підприємства, що користуються.

Серед інших відомостей, які складають предмет комерційної таємниці підприємства, можна виділити безпосередньо:

– важливі елементи систем безпеки, кодів і процедур доступу до інформаційних мереж і центрів;

– принципи організації охорони і захисту комерційної інформації і комерційної таємниці на підприємстві.

З відомостями такого роду постійно працюють, і часто вони знаходяться не в електронному вигляді, а в твердому (роздруковані). Тому виникає потреба в обмеженні доступу в приміщення, де вони обробляються та зберігаються.

1.2. Сучасні технології організації безпечного доступу на об’єкт

Найбільш часте застосування контроль доступу знаходить в офісних будівлях, готельних комплексах і невеликих готелях, бізнес-центрах, що здають в оренду свої приміщення безлічі компаній, магазинах і, звичайно ж, контрольно-пропускних пунктах (КПП). Службовці часто ходять з одного кабінету в інший і, природно, можуть забувати закривати двері, коли в приміщенні нікого не залишається. В цьому випадку контроль доступу є невід’ємною частиною загальної системи безпеки будівлі.

Крім цього СКУД може застосовуватися на території виробничих або торгових компаній для обмеження доступу персоналу в приміщення з цінними речами. Можливість проходу в них дістають тільки люди, що мають певний рівень довіри і відповідальності.

Використання контролю доступу також має місце на стоянках, парковках і в’їздах на територію. Зазвичай проводиться інтеграція засобів СКУД з автоматикою – шлагбаумами і коморами.

Контроль доступу допомагає людині виконувати рутинні завдання і по суті, як і будь-яка інша система безпеки, є інструментом в забезпеченні безпеки, тоді як людина приймає на себе управління в складних ситуаціях, котрі вимагають нестандартних рішень. Контроль переміщення людей або транспорту – це те, що входить в область дії системи, і залежно від наявних засобів і функцій дозволяє протидіяти несанкціонованому проходу, проїзду або проносу на територію, що охороняється, або з неї, вести реєстрацію людей і автомобілів, зберігаючи інформацію про те, коли людина прийшла / пішла, і на підставі цих даних створювати звіти для подальшого аналізу і обробки.

Такого роду автоматизація дозволяє здійснювати рух, реєстрацію людей і транспорту цілодобово, не задіюючи додаткові ресурси. В сукупності з іншими системами безпеки, такими як пожежна або охоронна сигналізація і відео спостереження, контроль доступу здійснює комплексну безпеку з централізованим управлінням.

Контроль доступу може бути автономною або мережевою системою. Мережева СКУД на відміну від автономної вимагає додаткового устаткування і програмного забезпечення, за рахунок чого його ціна вища. У випадку з автономним скуд всі дані (код карти, дані про власника, дозволений час проходу і так далі) зберігаються в пам’яті контролера і нікуди не передаються.

1.3. Огляд існуючих систем контролю доступу на об’єкт

Комп’ютери і сервери зі встановленим програмним забезпеченням – верхній рівень класичної мережевої системи контролю управління доступом (СКУД), вони управляють підключеними до них контролерами.

Контролер (контрольна панель) – це спеціалізований високо надійний комп’ютер. У ньому зберігається інформація про конфігурацію, режими роботи системи, список людей, які мають право входити в приміщення, а також їх права доступу в ці приміщення (коли і куди саме можна ходити).

У великих системах контролерів може бути декілька. У простих випадках мінімальний варіант контролера може бути вбудований в зчитувач.

Зчитувач є другою важливою ланкою в СКУД і підключається до контролера. Зчитувач це пристрій, який дозволяє прочитувати інформацію, записану на ідентифікаторі (картка, брелок). Цю інформацію він передає в панель, яка і ухвалює рішення про допуск людини в приміщення. Можна набудувати панель так, що вона запрошуватиме підтвердження ухваленого рішення у оперативного чергового.

Ідентифікатор (карта, брелок, біометрична ознака) має свій унікальний номер, якому приписаний деякий рівень доступу, відповідно до якого користувач має право проходу через ті або інші двері/турнікети в певні проміжки часу. Якщо ідентифікатор використовується як карта, її одночасно можна використовувати як пропуск з фотографією (нанести фотографію можна за допомогою спеціалізованого принтера карт) [4].

Виконавчі пристрої є “нижніми” елементами СКУД, це може бути електромеханічний або електромагнітний замок, турнікет, шлюз, шлагбаум, автоматичні ворота і так далі. Основна функція виконавчого пристрою – виконати команду контролера на блокування або розблокування проходу (проїзду).

Відразу треба відзначити, що застосування електронних СКУД не виключає участь людини в процесі управління. Найбільш ефективною є система, де електроніка, що є всього лише інструментом забезпечення безпеки, бере на себе виконання простих рутинних операцій, а людина виконує функції загального контролю і бере на себе управління в складних нестандартних ситуаціях (тобто займається тим, що у нього краще всього виходить).

Проте, сучасні СКУД дозволяють ефективно забезпечувати регулювання і контроль процесів доступу, і переміщення людей, машин, будь-яких інших об’єктів в зонах, що охороняються.

Залежно від складності, вони дозволяють: попереджати і протидіяти несанкціонованому проникненню осіб в заборонені приміщення, споруди і зони; виявляти факт проносу зброї, заборонених матеріалів і речовин; реєструвати факт проходу кожної людини, з прив’язкою до часу і додатковою службовою інформацією для подальшого аналізу і обробки.

Також в них можуть бути реалізовані можливості дистанційного керування процесом доступу людей і автотранспорту на територію і в кожне приміщення за рахунок застосування спеціалізованих виконавчих пристроїв (турнікети, шлюзи, шлагбауми і інші виконавчі пристрої СКУД). Автоматизація дій і реакцій дозволяє працювати 24 години на добу, не помилявшись при виконанні одноманітних рутинних операцій. Зрештою, підвищується загальна ефективність різних підсистем безпеки за рахунок їх інтеграції в єдиний комплекс з централізованим управлінням: СКУД, системи охоронно-пожежної сигналізації (ОПС), відео-нагляду, управління автоматикою будівель і споруд (інтегровані системи). Найбільш досконалими і ефективними СКУД є системи з комп'ютерним управлінням, в яких широкі можливості електроніки управління доповнюються практично необмеженим потенціалом програмного забезпечення.[9]

За способом управління СКУД діляться на: автономні, централізовані (мережеві) і універсальні.

Автономні системи призначені для управління одним або декількома виконавчими пристроями без обміну інформацією з центральним пультом (зазвичай комп'ютером) і без контролю з боку оператора.

Застосування таких систем цілком виправдане, коли потрібно контролювати прохід через одну – дві точки і немає необхідності вести онлайн моніторинг подій доступу. У таких системах часто сам контролер конструктивно об'єднується в одному корпусі зі зчитувачем.

Програмуються вказані контролери, як правило, або кнопкових панелей або за допомогою «мастер» – карт, що дозволяють заносити в пам'ять контролера нові карти і видаляти старі.

До переваг автономних систем відносяться невисока вартість, простота програмування системи, оперативність монтажу, зручність використання для невеликих об'єктів, що не мають спеціалізованих охоронних підрозділів.

Небагата функціональність породжує і свої недоліки – це незручність процесу програмування у разі великої кількості дверей і користувачів, відсутність можливості оперативної дії на процес проходу, відсутність

можливості обробки протоколу подій і отримання вибіркового звіту по заданих критеріях.

Централізовані (мережеві) системи – це величезний клас СКУД, головна особливість яких в тому, що вони мають можливість конфігурації апаратури і управління процесом доступу з комп'ютерних терміналів. Різні СКУД мають свої індивідуальні особливості і розрізняються по архітектурі, масштабу і можливостям, типу вживаних зчитувачі, ступені стійкості до злому і тому подібне [11].

Більшість мережевих СКУД зберігають багато переваг автономних систем, основне з яких – робота без використання комп'ютера, що управляє. Це означає, що при виключенні комп'ютера, що управляє, система фактично перетворюється на автономну. Контролери даних систем, так само як і автономні контролери, мають власний буфер пам'яті номерів карт користувачів і подій, що відбуваються в системі. Наявність в системі комп'ютера дозволяє службі безпеці оперативно втручатися в процес доступу і здійснювати управління системою в режимі реального часу.

Найважливішим елементом мережевих СКУД є програмне забезпечення. Воно відрізняється великою різноманітністю – від простих програм (які дозволяють додавати в базу даних нових користувачів і видаляти вибулих) для одного терміналу, що управляє, до складних програм з архітектурою клієнт-сервер.

Перевага мережевих систем полягають в зручності програмування, можливості прямого оперативного управління об'єктами, наочному відображення подій в режимі реального часу, можливості створення систем великого масштабу. Окремо потрібно відзначити широкі можливості інтеграції з іншими системами безпеки [18].

Список недоліків набагато скромніший – це необхідність навчання персоналу, вища вартість порівняно з автономними СКУД і вищими витратами на монтаж.

Універсальні системи є різновидом мережевих СКУД, вони здатні переходити в режим автономної роботи у разі виникнення відмов комп'ютера,

мережевого устаткування або обриву зв’язку. Такі системи забезпечують підвищений рівень надійності, проте в деяких системах при переході в автономний режим втрачається ряд важливих функцій, на що потрібно звертати особливу увагу. Це пов’язано з архітектурою систем і ступенем реалізації функцій на апаратному рівні (чим більше реалізовано апаратних функцій, тим вище ступінь автономності і надійності універсальної СКУД) [16].

1.4. Принцип роботи мережевої системи контролю доступу

Співробітники отримують в постійне користування електронний ключ у вигляді картки або брелока із записаним унікальним кодом (у випадку з відвідувачами їм видаються гостьові картки). Картки закріплюються за персоналом, а їх дані заносяться в базу даних на робочій станції. На місцях де необхідно організувати точку доступу, встановлюються зчитувачі, що розпізнають код карти і що передають його в контролер. Він порівнює дані з бази з прочитаним кодом, на підставі чого ухвалює рішення – відкривати прохід чи ні. У позитивному випадку подається команда відкриття на виконавчий пристрій, будь-то турнікет, шлагбаум або електромагнітний замок, після чого людина отримує доступ і може пройти. При відмові може бути задіяна охоронно-пожежна сигналізація (ОПС), блокування дверей або інші дії залежно від функціональності системи контролю доступу. Дані про того, хто і коли пройшов, фіксуються в базі даних. На їх підставі можна формувати звіти, що дозволяють вести облік робочого часу, виявляти порушників трудової дисципліни або простежувати шляхи проходження можливих зловмисників.

Головною перевагою мережевої СКУД є висока масштабованість, тобто здатність системи збільшувати свої можливості шляхом нарощування числа функціональних блоків, що виконують одні і ті ж завдання. Завдяки цьому вона підходить як для приватних будинків і маленьких офісів, так і для крупних готелів, ділових центрів і заводів. Перед системами безпеки і контролю доступу сьогодні стоять непрості завдання: вони повинні надійно захищати від небажаних проникнень на територію, що охороняється, але в той же час

безперешкодно пропускати своїх, бути зручними в роботі і доступними за ціною [9].

Сучасні СКУД – це інтегровані програмно-апаратні системи. Вони дозволяють в режимі реального часу управляти всіма дверима, шлагбаумами і турнікетами за допомогою одного ключового елементу. Це можуть бути електронні ключі з чіпом, радіо пульти, карточки-проксиміті і навіть прилади отримання біометричної інформації (відбитків пальців, долонь, будови сітківки ока і т.д.). Все це – ідентифікатори. Всіма перерахованими процесами управляє спеціалізоване програмне забезпечення. Під час роботи воно проводить швидкий збір, обробку і зберігання даних про перетини охоронних зон, а також надає масу додаткових можливостей: гнучку настройку пропускних режимів, організацію індивідуальних розкладів доступу для кожної особи, збір і аналіз відомостей про тривалість його перебування в тому або іншому приміщенні, реєстрацію спроб несанкціонованого доступу та ін.

Система контролю доступу і обліку робочого часу для невеликого офісу з одним входом. Поряд з дверима встановлюються зчитувачі на вхід і на вихід. Прикладаючи картку до зовнішнього зчитувача, співробітник відкриває двері. Час проходу записується в базу даних і служить відміткою про початок робочого дня. При виході в кінці дня прикладена картка відзначає кінець робочого дня. Проходи протягом робочого дня, наприклад, на обід, залежно від настройок програмного забезпечення, можуть відніматися з робочого часу, або не впливати на тривалість робочого дня. Співробітники з різних підрозділів можуть мати різні розклади доступу і робочого часу на день і тиждень. Всього може бути задане до 7 розкладів. Пам’ять контролера зберігає більше 4000 карт доступу і стількох же подій. У разі виключення комп’ютера система працюватиме автономно. Після відновлення зв’язку з комп’ютером всі дані про проходи будуть автоматично записані в базу даних. [10]

Мережева система контролю доступу і обліку робочого часу для офісу з двома входами. Для реєстрації карт співробітників використовується настільний зчитувач, що підключається до комп’ютера по *USB*.

Мережева система контролю доступу і обліку робочого часу для

устаткування прохідною невеликого підприємства. Комп'ютер, встановлений на прохідній, виконує функції зв'язку з контролером, турнікетом і фотоверифікації. Реєстрація карт співробітників і управління системою контролю доступу здійснюється за допомогою комп'ютера, встановленого у відділі кадрів [12].

1.5. Вимоги до виконавчих пристроїв СКУД

Вимоги до пристроїв ідентифікації накладють основні обмеження на зчитувачі. Зчитувачі повинні забезпечувати надійне прочитання коду з ідентифікаторів перетворення його в електричний сигнал і передачу на контролер.

Зчитувачі повинні бути захищені від маніпулювання шляхом перебору, підбору коду і радіочастотного сканування.

При введенні невірного коду повинне блокуватися введення на якийсь час, величина якого задається в паспортах на конкретні види зчитувачів. Час блокування повинен бути вибраний так, щоб забезпечити задану пропускну спроможність при обмеженні числа спроб підбору. При трьох спробах введення неправильного коду повинне видаватися тривожне сповіщення. Для систем, що працюють в автономному режимі, тривожне сповіщення передається на звуковий/світловий оповісник, а для систем, що працюють в мережевому режимі, – на центральний пульт з можливістю дублювання звуковим/світовим оповісником. Тривожне сповіщення повинне видаватися також при будь-якому акті вандалізму.

Конструкція, зовнішній вигляд і написи на ідентифікаторі та зчитувачі не повинні приводити до розкриття секретності коду.

Пристрої ідентифікації, аналогічно виконавчим пристроям, повинні бути захищені від впливу шкідливих зовнішніх чинників і вандалізму.

Виробник повинен гарантувати, що даний код ідентифікатора не повториться, або вказати умови повторюваності коду і міри по запобіганню використання ідентифікаторів з однаковими кодами.

Для автономних систем користувач повинен мати можливість змінити або

переустановити відкриваючий код в міру необхідності, але не менше 100 разів. Зміна коду повинна бути можлива тільки після введення коду, що діє. При виборі ідентифікаторів слід мати на увазі, що клавіатура забезпечує низький рівень безпеки; магнітні картки – середній; *Proximity*, Віганд – картки і електронні ключі *Touch Memory* – високий; біометричні – дуже високий рівень безпеки.

Вимоги до виконавчих пристроїв:

– пристрої повинні забезпечувати відкриття/закриття замкового механізму або пристрою загороди при подачі сигналу, від контролера, а також необхідну пропускну спроможність для даного об'єкту;

– параметри сигналу (напруга, струм і тривалість), що управляє, повинні бути вказані в стандартах і/або ТУ на конкретні види пристроїв загороди;

– величина напруги, що рекомендується, лінія 12 або 24 *В*. Проте, для деяких видів приводів виконавчих пристроїв (ворота, масивні двері, шлагбауми) допускається використовувати електроживлення від мережі 220/380 *В*;

– умисне пошкодження зовнішніх електричних сполучних ланцюгів не повинне приводити до відкриття пристрою загороди;

– у разі зникнення електроживлення в пристроях, повинна передбачатися можливість живлення від резервного джерела струму, а також аварійне механічне відкриття пристроїв загороди;

– аварійна система відкриття повинна бути захищена від можливості використання її для несанкціонованого проникнення;

– пристрої повинні бути захищені від впливу шкідливих зовнішніх чинників (електромагнітних полів, статичної електрики, нестабільної напруги живлення, пилу, вологості, температури і тому подібне) і вандалізму.

Вимоги до пристроїв контролю і управління доступом:

1) контролери, що працюють в автономному режимі, повинні забезпечувати прийом інформації від зчитувачів, обробку інформації і вироблення сигналів управління для виконавчих пристроїв.

2) контролери, що працюють в мережевому режимі, повинні забезпечувати:

– обмін інформацією по лінії зв’язку між контролерами і керівним комп’ютером або провідним контролером;

– збереження пам’яті, установок, кодів ідентифікаторів при обриві зв’язку з комп’ютером (провідним контролером), що управляє, відключенні живлення і при переході на резервне живлення;

– контроль ліній зв’язку між окремими контролерами і між контролерами і комп’ютером, що управляє.

Для гарантованої роботи СКУД відстань між окремими компонентами не повинна перевищувати величин, вказаних в паспортах (якщо не використовуються модеми).

Протоколи обміну інформацією і інтерфейси повинні бути стандартних типів. Види і параметри інтерфейсів повинні бути встановлені в паспортах і/або інших нормативних документах на конкретні засоби з урахуванням загальних вимог ГОСТ 26139.

Типи інтерфейсів, що рекомендуються:

– між контролерами *RS 485*;

– між контролерами *RS 485* і керуючим комп’ютером.

Вигляд і ступінь захисту повинні бути встановлені в паспортах на конкретні види засобів або систем. Відомості, приведені в технічній документації, не повинні розкривати секретність захисту.

Програмне забезпечення при необхідності належного бути захищено від несанкціонованого копіювання.

Програмне забезпечення повинне бути захищене від несанкціонованого доступу за допомогою паролів. Кількість рівнів доступу по паролях повинна бути не менше 2.

Рівні доступу за типом користувачів, що рекомендуються:

– перший ("адміністрація") – доступ до всіх функцій контролю і доступу;

– другий ("оператор") – доступ тільки до функцій поточного контролю;

– третій ("системщик") – доступ до функцій конфігурації програмного забезпечення, без доступу до функцій, що забезпечують управління пристроїв.

При введенні пароля на екрані дисплея не повинні відображатися знаки,

що вводяться. Число символів пароля повинне бути не менше 5.

Вимоги до електроживлення

Електропостачання засобів СКУД повинне здійснюватися від вільної групи щита чергового освітлення або від спеціально встановленого для цих цілей електрощита. Щит електроживлення, що встановлюється поза приміщенням, що охороняється, повинен розміщуватися в металевій шафі і повинен бути заблокований на відкриття.

Основне електроживлення СКУД повинне здійснюватися від мережі змінного струму частотою 50 *Гц* з номінальною напругою 220 *В*. СКУД повинні зберігати працездатність при відхиленнях напруги від -15 до +10 % і частоти до ±1 *Гц* від номінального значення.

Електроживлення окремих СКУД допускається здійснювати від інших джерел з іншими параметрами вихідної напруги, вимоги до яких встановлюються в нормативних документах на конкретні типи систем.

Засоби СКУД повинні в обов'язковому порядку мати резервне електроживлення при зникненні основного джерела електроживлення.

Перехід на резервне живлення і назад повинен відбуватися автоматично без порушення встановлених режимів роботи і функціонального стану СКУД. Номінальна напруга резервного джерела живлення повинна бути 12 або 24 *В*.

При використанні, як резервне джерело, живлення акумуляторної батареї, повинна забезпечуватися робота засобів СКУД не менше 8 годин.

1.6. Висновки до розділу

При проведенні аналізу розглянутих вище систем контролю доступу, основаних на класичних методах, було виявлено ряд недоліків:

1. Проблеми виникають із забороною подвійного проходу, функція якого є в більшості СКУД, контролюючих не тільки вхід, але і вихід. Ця функція призначена для того, щоб утруднити можливість передачі ключа-ідентифікатора іншій особі: якщо даним ключем вже скористалися при вході, ідентифікатор буде блокований до тих пір, поки контролер не зафіксує вихід його власника з

об’єкту. Проте в житті виникає безліч ситуацій, при яких чоловік, скориставшись ключем, не входить в приміщення або зволікає з входом. Ось в цьому випадку при повторному використанні ідентифікатора система відмовляє співробітникові у вході, оскільки по її відомостях він вже увійшов і знаходиться на об’єкті.

2. Певні проблеми можуть виникнути, якщо співробітник втратив, забув або десь залишив свій ідентифікатор. Якщо подібна ситуація відбулася поза об’єктом, що охоронявся, то питання входу можна без особливих проблем вирішити з охоронцем або оператором СКУД. Але якщо ключ загубився на роботі, а людина не може вийти з приміщення, що охороняється, і пропускає час, після якого виходи блокуються автоматично, то проблема може бути набагато складнішою.

3. Ідентифікатор може бути викрадений або дубльований, тоді безпека підприємства під загрозою. Тим більше, що дані системи не можуть забезпечувати захист об’єктів з підвищеними вимогами безпеки.

Виходом з даної ситуації може бути використання засобів біометричної ідентифікації. Оскільки в них ідентифікується не предмет (брелок, магнітна карта та ін.), а сама особа – носій біометричної інформації.

РОЗДІЛ 2
ПРИНЦИПИ РОБОТИ СИСТЕМ ВІДЕОСПОСТЕРЕЖЕННЯ

2.1. Структура і основні елементи систем відеоспостереження

Основне завдання системи відеоспостереження – отримання, запис і відтворення візуальної інформації про поточні події на об’єкті, що охороняється. Пристрої, представлені сьогодні на ринку обладнання для систем безпеки, дають можливість спроектувати систему відеоспостереження з хорошими технічними характеристиками, надійну і зручну в експлуатації.

Більшість систем відеоспостереження будується на базі локальної мережі. Це пов’язано з тим, що такий підхід відносно простий і звичний, перш за все, для користувача: управління елементами системи здійснюється через ПК, підключений до мережі. Проте як і раніше поширений «класичний» варіант реалізації на базі багатоканальних відеореєстраторів (рис. 2.1). Відеокамери розрізняються і за основними параметрами, і за конструктивними особливостями, і за принципом обробки сигналу.

Рис. 2.1 «Класичний» варіант системи відеоспостереження

2.2. Області застосування і огляд програмного забезпечення систем відеоспостереження

Контроль периметра великого об’єкта або стеження за потоком відвідувачів в супермаркетах – цим можливості *IP* -відеоспостереження аж ніяк не обмежуються. Широкий функціонал, простота інтеграції і гнучкість налаштувань дозволяють застосовувати мережеві камери для вирішення як звичних, так і нетривіальних завдань.

Кафе, бари, ресторани – для організації відеоспостереження за роботою кафе, барів і ресторанів використовуються *IP*-системи з функціоналом, подібним попередньому рішенню. Але перед ними ставляться ще й специфічні завдання, наприклад, повідомити адміністратора про прихід відвідувачів, обчислити час, проведений клієнтами в закладі.

Відстеження поведінкового фактора дозволяє визначити найбільш затребувані зони і використовувати цю інформацію в маркетингових цілях.

Додатковий бонус – можливість об’єднати в мережу відразу кілька закладів, що підвищує рентабельність бізнесу.

Майже будь-яка *IP* -система відеоспостереження підходить для контролю території заміського будинку, дачної ділянки або міської квартири.

Спеціальне програмне забезпечення орієнтоване на те, щоб відстежувати ситуацію і переглядати архіви з комп’ютера, планшета або смартфона. Інформація про виявлення руху в підконтрольній зоні надходить у вигляді повідомлення на телефон. Промислові та складські об’єкти – підвищити конкурентоспроможність складу, промислового підприємства або логістичного об’єкта в умовах масштабної конкуренції можна різними способами, і один з найефективніших – встановити камери і спеціальне програмне забезпечення *Macroscop*. Пристрої будуть відслідковувати рухомі об’єкти і відсилати тривожні повідомлення, проконтролюють в’їзд і виїзд автотранспорту по номерам.

Інтелектуальні функції ПЗ, наприклад розпізнавання осіб співробітників для фіксації проходу і організації СКУД, допоможуть налагодити дисципліну на підприємстві. Вкладення в таке рішення окупляться зростанням ефективності роботи, і компанія швидко випередить своїх конкурентів за кількістю клієнтів або товарообігу [3].

Офісні приміщення – практика показує, що встановлення відео-спостереження в офісі сприяє оптимізації бізнес-процесів і окупається досить швидко за рахунок підвищення працездатності кожного співробітника і налагодження дисципліни. Апарати стеження оперативно сповістять про наявність людей в позаурочний час, нададуть повну інформацію про те, що відбувається в стінах кожного підконтрольного кабінету і допоможуть вирішити будь-який конфлікт, особливо якщо доповнити обладнання мікрофонами для запису звуку.

Автостоянки – з не меншою педантичністю, ніж за будинком, квартирою або здоров'ям, сьогодні люди стежать за безпекою власного транспорту, тому присутність мережевого спостереження на автостоянках буде незайвим. Крім банального контролю наявності людей з метою запобігання актам вандалізму або викрадення, система вміє фіксувати автономери. Ці дані знадобляться для вирішення конфліктних ситуацій, що виникають в результаті пошкодження транспортного засобу або ДТП.

Програмне забезпечення розроблене для можливості віддаленого перегляду і управління цифровими системами відеоспостереження. З її допомогою можна здійснити таке завдання, як управління відеосерверамі і *IP*-камерами з будь-якої відстані за наявності інтернету, а також управління купольними і поворотними камерами. Вона наділена всіма функціями, як і *AXIS CameraRecorder*, але принцип роботи побудований накліент – серверному додатку. Для установки на персональний комп'ютер оператора достатньо клієнтського додатка, що б забезпечити віддалене відеоспостереження. У стандартній конфігурації *AXIS CameraStationClient* можна переглядати і записувати від чотирьох до десяти відеоканалів. Якщо буде потрібно збільшити

кількість каналів, то це можна здійснити за допомогою покупки додаткових ліцензій *AXIS CameraStationClient*.

Компанія «*AXIS*» є однією з провідних виробників цифрових систем відеоспостереження на світовому ринку, з цієї причини інші компанії, що виробляють аналогове обладнання систем безпеки передбачають необхідність поєднання з продукції з обладнанням *AXIS*. Тобто обладнання систем безпеки інших виробників підтримує протоколи, що застосовуються мережевим обладнанням *AXIS*. Цифрові входи і релейні виходи *IP* – камер і відеосерверів *AXIS* дають можливість отримання зображення і управління обладнанням системи безпеки і системи управління об'єкту, що охороняється.

Для того, що б зареєструвати в мережі пристрої для цифрового відеоспостереження – *IP* – камери і відеосервери, можна скористатися кількома способами.

Перше – це привласнення *IP* – адреси вручну із застосуванням *ARP* і *Ping*, вручну через *AXIS IP Utility* або ж автоматично через сервіс *AXIS InternetDynamic DNS*. Дана служба здійснює діяльність на серверах компанії для відстеження її динамічного *IP* – адреси. Що б отримати доступ до відеокамери або відеосерверів зовсім не обов'язково запам'ятовувати *IP* адреси, досить ввести в адресному рядку браузера *http*: // *axisXXXXXX*. *axiscam*. *net*, де XXXXXX – є серійним номером камери або відео-сервера.

Деякі різновиди *IP* – камер *AXIS* мають функції бездротового підключення до *IP* – мережі. Це моделі *AXIS 206 W* і *AXIS 207 W IS* 206 *W* працює за стандартом *IEEE* 802.11 *b* (*WEP*), *AXIS* 207 *W* підтримує поліпшену систему бездротової передачі даних *WPA* / *WPA* 2.

Ця функція можлива в застосуванні з умовою багаторівневої аутентифікації. Тобто, доступ до бездротової мережі можливо тільки при правильному введенні всіх паролів. Кожній камері присвоєно окремий пароль.

"Інтелект" від компанії *ITV* Компанія – *ITV* більш ніж відома в країнах СНД, а на світовому ринку вона розвивається під брендом *AXXON*. Має в портфоліо велику кількість інсталяцій самого різного масштабу – від невеликих офісів до систем "Безпечне місто".

Деякі вендори прийшли до систем через аналогове відео, інші спочатку розробляли ПО виключно для мережевих систем відеоспостереження. Кожен продукт цікавий і має свої особливості, які в підсумку впливають на відмовостійкість і надійність системи За всіма вказаними параметрами "Інтелект" виглядає виграшно – як в плані набору базового функціоналу, так і з точки зору інтелектуальних модулів. Виняток становить підтримка "гарячої" заміни при виході з ладу основного сервера, що може бути критично при спостереженні за об'єктом особливої важливості, а також вимагати постійної присутності персоналу. ПО "Інтелект" від компанії *ITV* є безумовним лідером за кількістю інтеграцій з обладнанням і ПО сторонніх виробників – як з *IP* -відео-камерами, системами ОПС і СКД, так і з касовими програмами, що відкриває широкі можливості з побудови комплексних (інтегрованих) систем безпеки, в тому числі і для вертикальних ринків. «Інтелект» – багатофункціональна відкрита програмна платформа, призначена для створення комплексних систем безпеки будь-якого масштабу. Завдяки «Інтелекту» комплекс різних систем безпеки перетворюється в єдине інформаційне середовище, в якій реалізовані функції обробки і інтелектуального аналізу інформації, що володіє здатністю гнучко реагувати на різні події. А завдяки модульній архітектурі замовник може вибирати саме ті функції, які потрібні для побудови ефективної системи безпеки конкретного об'єкта – таким чином, отримуючи систему з оптимальним набором функцій і мінімальними витратами. Одне з ключових переваг «Інтелекту» – спеціалізовані галузеві рішення, призначені для конкретних напрямків економіки та бізнесу, а також для захисту державних і інфраструктурних об'єктів.

SecurOS від компанії *ISS – ISS* є однією з компаній, які стояли біля витоків появи і розвитку комп'ютерних систем безпеки в Україні. Її співробітники багато зробили для впровадження понять "комплексна" і "інтегрована" системи безпеки. Характеристики та функціонал системи *SecurOS* дуже гнучкі: наприклад, є можливість відобразити інтерфейс, адаптований за зовнішнім виглядом під *DVR*, що дозволяє істотно скоротити час навчання персоналу, який звик працювати з реєстраторами або відеомагнітофонами. У лінійці продукції

компанії *ISS* можна знайти рішення для широкого спектра об'єктів, в тому числі і рішення, повністю готові до монтажу на об'єкті. *SecurOS* широко поширений, і різні інтелектуальні модулі, розроблені в *ISS*, використовуються на багатьох промислових підприємствах: наприклад, для розпізнавання номерів залізничних вагонів, що дозволяє істотно підвищити прозорість обороту рухомого складу. Успішні інсталяції, в тому числі участь в проектах "Безпечне місто", говорять про високу якість системи.

VideoNet від Корпорації СКАЙРОС – популярна на українському ринку система *VideoNet* займає помітну його частку і розвивається в бік розширення інтелектуальних детекторів руху. В систему інтегрований сторонній математичний апарат для розпізнавання номерів автотранспорту та осіб людей. Реалізовано алгоритми визначення напрямку руху і перетину лінії, вбудовані фільтр корекції коливань джерела зображення *VNImageStabilization* ™ і фільтр корекції оптичних "бочкоподібних" спотворень *VNLensCorrection* ™, що може бути дуже корисно на споруджуваних або промислових об'єктах для поліпшення складних умов спостереження і експлуатації системи. Компанією також розроблений власний кодек *DVPack* 2 ™, який дозволяє передавати на клієнтські робочі місця відео в необхідному дозволі без зайвої завантаження локальної мережі і застосовувати відеоаналітіка без розпакування відеопотоку, що істотно заощаджує ресурси сервера. У систему інтегровано досить велика кількість сторонніх систем, що робить *VideoNet* підходящої для вирішення широкого кола завдань.

XprotectCorporate откомпанії *MilestoneSystems* – компанія *MilestoneSystems* (Данія) однією з перших в світі звернула увагу на *IP* -відеоспостереження, що визначило сьогоднішнє лідерство компанії на світовому ринку. Програмний продукт є платформою для розподіленої комплексної системи безпеки та має найбільшу кількість інтегрованих брендів і кінцевих *IP* -відеопристроїв, причому швидкість інтеграції дуже швидка, включення продукту в список підтримуваних часто випереджає доступність самого продукту для замовлення.

XProtectCorporate – це ефективна програма для управління *IP*-відео (*VMS*), створена спеціально для розгортання в великомасштабних зонах з високим

рівнем безпеки. Єдиний інтерфейс керування дозволяє ефективно керувати системою (включаючи всі камери і пристрої системи безпеки), незалежно від її розміру і того, розміщена вона на одному об’єкті або на декількох.

Для систем, що вимагають максимальної ситуаційної обізнаності та точної реакції на події, *XProtectCorporate* надає інтерактивні карти, пов’язані з сигнальними пристроями, а також підтримку *XProtect ® SmartWall*. *XProtectCorporate* гарантує найвищу надійність для систем з високим ступенем безпеки. Підтримка сховища *EdgeStorage* в поєднанні з резервними серверами для запису і резервними серверами управління забезпечує безперервність відеоархіву.

2.3. Інтегровані системи безпеки

Тенденції сучасного розвитку систем безпеки нерозривно пов’язані з процесами широкої автоматизації та інтеграції, які стосуються не тільки систем безпеки, а й усіх інших систем, призначених для автоматизації управління життєзабезпеченням і функціонуванням житлового будинку, офісу, підприємства або будь-якого іншого об’єкта.

Логічним розвитком такої інтеграції стало створення інтегрованих систем безпеки (ІСБ) з широкими функціональними можливостями, що дозволяють автоматизувати також управління інженерними системами будівлі або об’єкта.

Основою таких ІСБ служить єдина апаратно-програмна платформа, що представляє собою автоматизовану систему управління (АСУ) з багаторівневою мережевою структурою, що має загальний центр управління на базі локальної комп’ютерної мережі та містить лінії комунікацій, контролери прийому інформації, керуючі контролери та інші периферійні пристрої, призначені для збору і обробки інформації від різних датчиків (в тому числі від сповіщувачів пожежної та охоронної сигналізації), а також для управління різними засобами автоматізації (оповіщення, протипожежна автоматика і пожежогасіння, інженерні системи і т.д.).

Інтегрована система безпеки (ІСБ) – сукупність техниче ських засобів (двох або більше взаємопов'язаний них АС), призначених для по будови систем охоронної, пожежної сигналізації та оповіщення, управління протипожежною автоматикою, контроля і управління доступом і систем телевізійного спостереження, які мають технічну, інформаційну, програмну та експлуатаційну сумісність так, що цю совокупність можна розглядати як єдину АС [5].

З цього визначення також випливає, що ІСБ це система, що забезпечує захист від декількох видів загроз. В даному вище визначенні – ІСБ призначена для захисту від пожежі (пожежна сигналізація, оповіщення, протипожежна автоматика) і від кримінальних загроз (охоронна сигналізація, контроль доступу, охоронне телебачення).

Сучасні ІСБ будуються на основі ієрархічної мережевої структури, в яку входять комп'ютерні мережі, а також локальні мережі різного рівня складності спеціальних обчислювальних пристроїв – контролерів. У ній можна виділити чотири рівні мережевої взаємодії.

Перший (верхній) рівень являє собою комп'ютерну мережу типу клієнт / сервер на основі мережі *Ethernet*, з протоколом обміну *TCP* / *IP* і з використанням мережевих операційних систем. Цей рівень забезпечує зв'язок між сервером і робочими станціями операторів. Управління ІСБ на верхньому рівні забезпечується за допомогою спеціалізованого програмного забезпечення (ВПЗ). Для невеликих об'єктів можливе використання для управління ІСБ одного комп'ютера. На верхньому рівні також забезпечується зв'язок і управління віддаленими об'єктами.

Сучасні можливості комп'ютерних мереж дозволяють передавати інформацію по різних каналах зв'язку, тим самим на основі ІСБ можна створювати системи моніторингу безпеки віддалених об'єктів.

Другий рівень – рівень локальних контролерів, основних компонентів управління ІСБ. Кожен локальний контролер повинен забезпечувати виконання основних функцій в своїй зоні контролю, навіть при порушенні зв'язку з верхнім рівнем ІСБ. Для зв'язку між однорідними контролерами (горизонтальний рівень зв'язку) використовується інтерфейс *RS* 485 або інші інтерфейси, призначені для

побудови мереж промислового рівня з гарною перешкодозахищеність і достатньою швидкістю обміну даними. Зв'язок між другим і верхнім рівнем (вертикальний рівень зв'язку) може забезпечуватися через один з мережевих контролерів, за допомогою підключення його до сервера ПО АРМ ІСБ через стандартний порт ПЕОМ. У контролерах деяких ІСБ можливий прямий вихід на перший рівень в протоколі *TCP*/*IP*.

Третій рівень – рівень адресних мережевих пристроїв, які підключаються до кожного контролера другого рівня. Тут, як правило, застосовується інтерфейс *RS* 485. Кількість мережевих пристроїв, що підключаються до одного контролера, може бути до 256. Номенклатура адресних мережевих пристроїв досить різноманітна, від простих розширювачів для підключення радіальних ШС до складних контролерів третього рівня, наприклад, пристроїв управління пожежогасінням або модулів підключення адресно- аналогових пожежних сповіщувачів.

Четвертий рівень – сповіщувачі і сповіщувачі ОПС, зчитувачі і виконавчі пристрої СКУД, датчики і пристрої управління технологічним обладнанням та ін. Тут, як правило, застосовуються нестандартні спеціалізовані інтерфейси і протоколи.

Технічні можливості ІСБ дозволяють визначити подальші перспективи їх розвитку – інтеграція з іншими системами автоматизації та розширення видів і кількості загроз, захист від яких забезпечується за допомогою ІСБ.

Тенденція подальшої інтеграції – об'єднання ІСБ з системами автоматизації і управління інженерними системами будівлі або об'єкта пов'язана з появою терміна – «інтелектуальний будинок». «Інтелектуальний будинок» – це комплекс проектних, організаційних, інженерно-технічних, програмних рішень, спрямованих на створення єдиної інформаційно-керуючої інфраструктури, що забезпечує гнучку та ефективну технологію обслуговування будівлі (об'єкта) і найбільш повно відповідає потребам його власників і орендарів з дотриманням сучасних вимог забезпечення безпеки.

Основне призначення системи «інтелектуального будинку» – забезпечення ефективності функціонування всіх інженерно-технічних систем,

енергозбереження, запобігання, виявлення і оперативне усунення будь-яких екстремальних ситуацій, що виникають в процесі експлуатації будівлі, максимально знижуючи наслідки можливого шкоди.

Термін «інтелектуального будинку» в більшій частині застосовується до житлових і офісних будівель. Інтеграція систем для виробничих і промислових об’єктів дає можливість побудови комплексів, в яких автоматизація виробничого процесу або основного функціонального призначення об’єкта тісно пов’язана із забезпеченням безпеки, як власне об’єкта, так і людини від різних видів загроз, які можуть виникнути на об’єкті в результаті його функціонування. Взаємозв’язок з системами життєзабезпечення, в цьому випадку дозволяє ефективно і економічно виконувати функціональні завдання. Такі системи, по суті, являють собою повноцінні автоматизовані системи управління функціонуванням, життєзабезпеченням і безпекою об’єкта (АСУ ФЖБ).

Процес створення системи безпеки об’єкта включає в себе ряд етапів, основні з яких це проектування, монтаж, пуско-налагоджувальні роботи, здача в приймання замовнику.

Кожен об’єкт, на якому створюється система безпеки, є унікальним, тому кожна проектована система являє собою продукцію одиничного виробництва, створювану заново для кожного конкретного об’єкта.

Стандарт встановлює порядок розробки, узгодження і затвердження технічного завдання, технічної документації, а також порядок виготовлення, контролю, монтажу, приймання та здачі в експлуатацію виробів одиничного виробництва і їх складових частин, остаточне складання, налагодження, випробування і доведення яких можуть бути проведені тільки на місці експлуатації в складі конкретного виробничого об’єкту.

Найважливішу роль при створенні системи грає процес проектування, так як на етапі проектування закладаються всі необхідні якісні характеристики системи. При проектуванні важливим питанням є вибір технічних засобів ІСБ, з яких буде створюватися система.

Тут під технічними засобами ІСБ розуміються – технічні вироби (продукція серійного виробництва, спеціально призначена для побудови ІСБ), а

також система в цілому, як продукція одиничного виробництва, створювана для кожного об’єкта шляхом проектування, монтажу, налагодження та здавання в експлуатацію, функціональним призначенням якої є забезпечення безпеки.

ІСБ в будь-якому випадку є складною технічноу системоу і при її створенні доводиться використовувати різне обладнання, як за функціональним призначенням, так і обладнання різних виробників.

При цьому завжди постає завдання сумісності обладнання. Причому вона включає в себе дві складові.

Перша – це завдання забезпечення взаємодії обладнання різних підсистем, об’єднаних в ІСБ. Друга – сумісність обладнання різних виробників. Ці завдання повинні бути вирішені на етапі проектування ІСБ і можуть бути оптимізовані в рамках вибору способу (платформи) інтеграції.

Принципи проектування ІСБ багато в чому визначаються способом інтеграції, який можна розбити на чотири основні рівні (платформи інтеграції):

– інтеграція на проектному рівні (проектна платформа) – об’єднання різнорідного устаткування, спеціально не призначеного для побудови ІСБ, тільки на етапі проектування системи;

– інтеграція на програмному рівні (програмна платформа) – об’єднання облад-нання різних виробників, на базі спеціально розробленого для інтеграції програмного продукту і управління системою на базі ПЕОМ загального призначення або ЛВС ПЕОМ;

– інтеграція на апаратно-програмному рівні (апаратно програмна платформа) – об’єднання обладнання та програмного продукту єдиного виробника і управління системою на базі ПЕОМ загального призначення або ЛВС ПЕОМ;

– інтеграція на апаратному рівні (апаратна платформа) – об’єднання обладнання та програмного продукту єдиного виробника і управління системою без використання ПЕОМ загального призначення, на основі спеціалізованих високопродуктивних контролерів і ЛВС на їх основі.

Особливо слід відзначити інтеграцію в ІСБ підсистеми відеоспостереження. Причому слід, перш за все, розглядати цифрові технології в

СОТ, як найбільш перспективні. Особливості інтеграції СОТ пов'язані з тим, що для передачі та обробки відеоданих в цифрових СОТ потрібні значні обчислювальні й інформаційні ресурси, тому реалізація цифрових СОТ в ІСБ можлива тільки на верхньому рівні управління на базі ПЕОМ або ЛВС ПЕОМ.

Рівні інтеграції:

1. Інтеграція на проектному рівні.

Об'єднання систем проводиться на етапі проектування системи для кожного конкретного об'єкта. Робота проводиться проектно-монтажними фірмами, які називають себе «системними інтеграторами». Як правило, в цьому випадку, застосовуються різнорідні підсистеми (продукція) різних виробників, які не придатні спеціально для взаємної інтеграції.

Об'єднання (інтеграція) цих систем здійснюється шляхом установки устаткування управління підсистемами в загальному приміщенні – центральному пункті управління. Взаємодія між підсистемами здійснюється на рівні операторів підсистем, тобто без автоматизації.

Очевидно, що це мінімальний рівень інтеграції, йому притаманні певні недоліки («людський фактор», різнорідність апаратури, складність обслуговування, паралельність прокладаються комунікацій, відсутність автоматизації і т.д.) і його не можна вважати в даний час перспективним, хоча є ряд фірм, які пропонують свої готові і перевірені проектні рішення.

Оптимальним підходом в цьому випадку, слід вважати розроблену фірмою – проектувальником власну проектну методологію побудови систем.

2. Інтеграція на програмному рівні.

У цьому випадку роль об'єднання підсистем грає спеціальне *спеціальне* програмне забезпечення (СПЗ) – програмний пакет, розроблений і поставляється як самостійний продукт (програмна продукція серійного виробництва, спеціально призначена для інтеграції технічних підсистем). Таке СПЗ призначене для функціонування в апаратної середовищі, як правило, в локальній мережі ПЕОМ загального призначення, яка представляє собою верхній рівень ІСБ. Сполучення з апаратною частиною підсистем нижнього рівня здійснюється за допомогою програм-драйверів, що розробляються спеціально для підтримки

конкретних засобів інших виробників. Зв’язок з апаратними засобами здійснюється за допомогою стандартних портів ПЕОМ.

Подібна побудова ІСБ має ряд позитивних сторін. Це можливість на програмному рівні, використовуючи всі можливості сучасних комп’ютерних технологій, створювати високоякісні багатофункціональні програмні системи. Можливість інтеграції з апаратними засобами інших виробників (при наявності відповідного драйвера і відповідних інтерфейсів обміну даними в самих застосовуваних засобах).

З іншого боку, це породжує і певні недоліки – необхідність розробки драйверів для кожного застосовуваного апаратного засобу. При цьому не завжди розробник апаратного засобу надає протоколи обміну даними. Навіть, якщо протоколи відкриті і документовані, в них можуть бути закладені обмежені можливості, що не дозволяють оптимальним чином забезпечити поєднання. Крім того, фірма розробник програмної системи, поставляючи тільки свій програмний продукт, не може в цьому випадку в повному обсязі гарантувати роботу всієї системи в цілому.

3. Інтеграція на апаратно-програмному рівні.

В цьому випадку апаратні і програмні засоби розробляються в рамках єдиної системи. Це дозволяє досягти оптимальних характеристик, так як вся розробка зосереджена, як правило, в одних руках і система як закінчений продукт поставляється з повною гарантією виробника.

Загальним недоліком наведених вище способів інтеграції є використання на верхньому рівні управління ІСБ персональних комп’ютерів загального призначення. Відомо, що ПЕОМ і базове ПЗ загального призначення (операційні системи, системи управління базами даних і ін.) Призначене в основному для офісного і домашнього вжитку. Вони володіють зайвою функціональністю (мультимедійні, ігрові та інші можливості побутових і офісних ПЕОМ) і недостатньою надійністю для вирішення завдань автоматизації управління системами, особливо системами безпеки [5].

Для використання в ІСБ необхідно застосовувати спеціалізовані промислові ПЕОМ і відповідне спеціалізоване базове ПЗ. Однак вартість такого рішення істотно вище.

4. Апаратна платформа інтеграції.

Апаратна платформа інтеграції – відносно новий напрямок розвитку принципів побудови ІСБ. При розробці даного напрямку ставилося завдання усунення загальної нестачі інших методів інтеграції, тобто відмова від використання в ІСБ на всіх рівнях ПЕОМ загального призначення.

Апаратний спосіб інтеграції – на основі обладнання без участі ПЕОМ, забезпечує максимальну надійність і швидкодію системи.

2.4. Можливості використання відеокамер в сучасних біометричних методах ідентифікації

Яким би зручним і досконалим не був ключовий елемент, завжди потрібно враховувати людський чинник. Будь-який ключ, пульт, електронну картку можна забути удома, на роботі або просто втратити. І тоді наша безпека під загрозою. Але і на цей випадок є рішення – використання блоків-зчитувачів біометричної інформації людини: відбитків пальців, будови сітківки ока, тембру голосу і так далі. Серед них найбільш поширені дактилоскопічні пристрої. Вони дозволяють зберігати інформацію про всі десять пальців рук кожного користувача і застосовувати для входу малюнок будь-якого з них. Для цього досить провести або прикласти палець до чутливого елемента. Системи доступу, засновані на біометричній інформації, забезпечують найвищий клас захисту, і тому найбільш затребувані в організаціях з надзвичайно серйозними вимогами до безпеки, наприклад, в банках.

Біометричний захист ефективніший порівняно з такими методами, як використання смарт-карт, паролів, *PIN*-кодів і тому подібне, оскільки біометрія дозволяє ідентифікувати людину, а не пристрій. Традиційні методи захисту не виключають можливості втрати або крадіжки інформації, унаслідок чого вона стає доступною незаконним користувачам. Відбиток же пальця або малюнок

радужної оболонки унікальні, вони завжди з вами, їх майже неможливо відняти і не можна забути.

Біометрія – унікальна, вимірна характеристика людини для автоматичної ідентифікації або верифікації. Термін «автоматично» означає, що біометричні технології повинні розпізнавати або верифікувати людину швидко і автоматично, в режимі реального часу. Ідентифікація за допомогою біометричних технологій припускає порівняння раніше внесеного біометричного зразка з біометричними даними, що знову надаються.

В даний час існує безліч методів біометричної ідентифікації, які можна розділити на дві великі групи: статистичні і динамічні.[28]

– статистичні та динамічні методи ідентифікації;

– статистичні методи;

– ті, що ґрунтуються на фізіологічній (статистичній) характеристиці людини, тобто унікальній властивості, даній йому від народження і невід'ємному від нього.

Динамічні методи ґрунтуються на поведінковій (динамічній) характеристиці людини, тобто побудовані на особливостях, характерних для підсвідомих рухів в процесі відтворення якої-небудь дії.

До динамічних методів відносять наступні методи:

– по рукописному почерку. Для цього методу використовується підпис людини (іноді написання кодового слова). Цифровий код формується по динамічних характеристиках написання, тобто будується згортка в яку входить інформація по графічних параметрах, тимчасовим характеристикам нанесення підпису і динаміки натиску на поверхню і так далі;

– по клавіатурному почерку. Метод аналогічний вищеописаному, але замість підпису використовується кодове слово. Основна характеристика по якій будується згортка – динаміка набору кодового слова;

– по голосу. Існує багато способів побудови коду ідентифікації по голосу, як правило, це різні поєднання частотних і статистичних характеристик голосу.

Статичні методи:

– по відбитку пальця. Найпоширеніший метод біометричної ідентифікації,

в основі якого лежить унікальність для кожної людини малюнка папілярних узорів на пальцях. Зображення відбитку пальця, отримане за допомогою спеціального сканера, перетвориться в цифровий код (згортку) і порівнюється з раніше введеним шаблоном (еталоном) або набором шаблонів (у разі ідентифікації);

– по розташуванню вен на тильній стороні долоні. За допомогою інфрачервоної камери прочитується малюнок вен на тильній стороні долоні або грона руки, отримана картинка обробляється, і за схемою розташування вен формується цифрова згортка.

По сітківці ока. Вірніше, це спосіб ідентифікації по малюнку кровоносних судин очного дна. Для того, щоб малюнок став видний, людині треба подивитися на видалену світлову крапку, і очне дно, що підсвічується таким чином, сканується спеціальною камерою.

За формою особи. У даному методі ідентифікації будується два або три мірний образ обличчя людини. За допомогою камери і спеціалізованого програмного забезпечення на зображенні виділяються контури очей, брів, носа, губ і так далі обчислюються відстань між ними. За цими даними будується образ, що перетворюється в цифрову форму для порівняння.

По термограмі особи. У основі даного методу лежить унікальність розподілу на обличчі артерій, що забезпечують кров'ю шкіру і що виділяють тепло. Для отримання зображення використовуються спеціальні камери інфрачервоного діапазону.

Всі біометричні системи працюють практично за однаковою схемою. По-перше, система запам'ятовує зразок біометричної характеристики (це і називається процесом запису). Під час запису деякі біометричні системи можуть попросити зробити декілька зразків для того, щоб скласти найбільш точне зображення біометричної характеристики. Потім отримана інформація обробляється і перетворюється в математичний код [9].

2.5. Висновки до розділу

Системи контролю і управління доступом (СКУД) міцно зайняли своє місце в переліку технічних систем безпеки, пропонованих на ринку.

Більш того, темпи зростання продажів устаткування СКУД складають 15%, а інших систем охорони в два рази менше – 7%. Будь-яка СКУД призначена для того, щоб автоматично пропускати тих, кому це належить, і не пропускати тих, кому це заборонено, контролюючи тим самим переміщення співробітників і відвідувачів на території підприємства.

Всі СКУД діляться на два типи: програмні і контролерні. Відмінність їх в тому, що в контролерних СКУД інформація про призначені картах і проходах спочатку закладається в контролер, і тільки потім переноситься на комп'ютер. У програмних же СКУД безпосередній зв'язок обладнання з комп'ютером дає можливість отримувати і обробляти інформацію відразу ж в момент її надходження.

Таким чином, в даний час системи контролю та управління доступом (СКУД) міцно зайняли своє місце в переліку технічних систем безпеки, пропонованих на ринку. Разом з охоронно-пожежною сигналізацією та системами телевізійного спостереження вони утворюють базу для інтеграції систем безпеки будівель в єдиний комплекс.

Так само невід'ємним елементом інтегрованої системи є система відеоспостереження.

Основне завдання системи відеоспостереження – отримання, запис і відтворення візуальної інформації про поточні події на об'єкті, що охороняється. Пристрої, представлені сьогодні на ринку обладнання для систем безпеки, дають можливість спроектувати систему відеоспостереження з хорошими технічними характеристиками, надійну і зручну в експлуатації.

Контроль периметра великого об'єкта або стеження за потоком відвідувачів в супермаркетах – цим можливості *IP* -відеоспостереження аж ніяк не обмежуються. Широкий функціонал, простота інтеграції і гнучкість

налаштувань дозволяють застосовувати мережеві камери для вирішення як звичних, так і нетривіальних завдань.

Інтегрована система безпека (ІСБ) – сукупність техниче ських засобів (двох або більше взаємопов'язаний них АС), призначених для по будови систем охоронної, пожежної сигналізації та оповіщення, управління протипожежної автоматикою, контролю та управління доступом і систем телевізійного спостереження, які мають технічну, інформаційну, програмну та експлуатаційну сумісність так, що цю сукупність можна розглядати як єдину АС. На нашому підприємстві використовуються електронні прохідні компанії *PERCo*, яка спеціалізується на серійному виробництві обладнання і систем безпеки власної розробки з 1988 року.

Електронні прохідні – це готові рішення, які дозволяють найпростішим і зручним способом організувати систему контролю доступу в установах і на підприємствах. Електронні прохідні виглядають як турнікети, але включають в себе всю електроніку систем контролю доступу. На нашому підприємстві використовуються електронні прохідні серії *KT* 02.3.

Також використовується система відеоспостереження *BEWARD*. Головною особливістю є те, що ПЗ здатне адаптуватися під конкретні вимоги, а точніше *ip* камери моделей *BD* 4330 *DS* і *N* 630 дуже добре інтегруються з системою безпеки *PERCO*, використовуваної в університеті. Використання даних моделей відеокамер також обгрунтовано найкращим співвідношенням ціни і якості.

Таким чином, використовувана система безпеки відповідає сучасним тенденціям, але не варто забувати і про перспективи розвитку інтегрованих систем безпеки.

РОЗДІЛ 3
ОПИСАННЯ ПРОГРАМНОЇ РЕАЛІЗАЦІЇ ІНТЕГРАЦІЇ СИСТЕМИ КОНТРОЛЮ УПРАВЛІННЯ ДОСТУПОМ З СИСТЕМОЮ ВІДЕОСПОСТЕРЕЖЕННЯ

3.1. Описання існуючої СКУД

На підприємстві використовуються електронні прохідні компанії *PERCo*, яка спеціалізується на серійному виробництві обладнання і систем безпеки власної розробки з 1988 року.

Вся продукція *PERCo* – турнікети, електромеханічні замки, електронні прохідні, системи контролю доступу та комплексні системи безпеки – випускається серійно. Устаткування і системи безпеки *PERCo* проходять необхідні ресурсні та кліматичні випробування, випробування на електробезпеку, електромагнітну сумісність, пожежну безпеку.

Сучасні виробничі потужності і застосування інноваційних технологій дозволяють *PERCo* розробляти, випускати і успішно представляти на ринку широкий асортимент товарів, що відповідають світовим стандартам в галузі обладнання і систем безпеки.

Електронні прохідні – це готові рішення, які дозволяють найпростішим і зручним способом організувати систему контролю доступу в установах і на підприємствах. Електронні прохідні виглядають як турнікети, але включають в себе всю електроніку систем контролю доступу.

Електронна прохідна *PERCo – KT* 02.3 (далі – ЕП) призначена для організації однієї двосторонньої точки проходу на територію підприємства.

Контроль доступу через ЕП здійснюється оператором за допомогою пульта дистанційного керування, що входить в комплект поставки або, після додаткової настройки з використанням ПЗ *PERCo – S* -20, за безконтактними картками доступу.

Склад:

– турнікет;

– вбудований контроллер СКУД;

– два вбудованих зчитувача безконтактних карт доступу (*HID* / *EM – Marin*);

– пульт дистанційного керування;

– програмне забезпечення *PERCo – SL* 01 «Локальне ПЗ».

Залежно від завдань підприємства вона може працювати як:

– самостійна система контролю доступу на 1 точку проходу;

– частина системи контролю доступу, яка обслуговує декілька точок проходу;

– автономний турнікет, керований оператором від пульта дистанційного керування.

Електронна прохідна КТ02.3 має можливість прямого підключення до комп'ютера або до локальної обчислювальної мережі підприємства (мережі *Ethernet*) для введення даних і отримання звітів.

У стандартному комплекті Електронної прохідної поставляється безкоштовне програмне забезпечення *PERCo – SL* 01 для організації набору функцій системи контролю доступу (рис. 3.1).

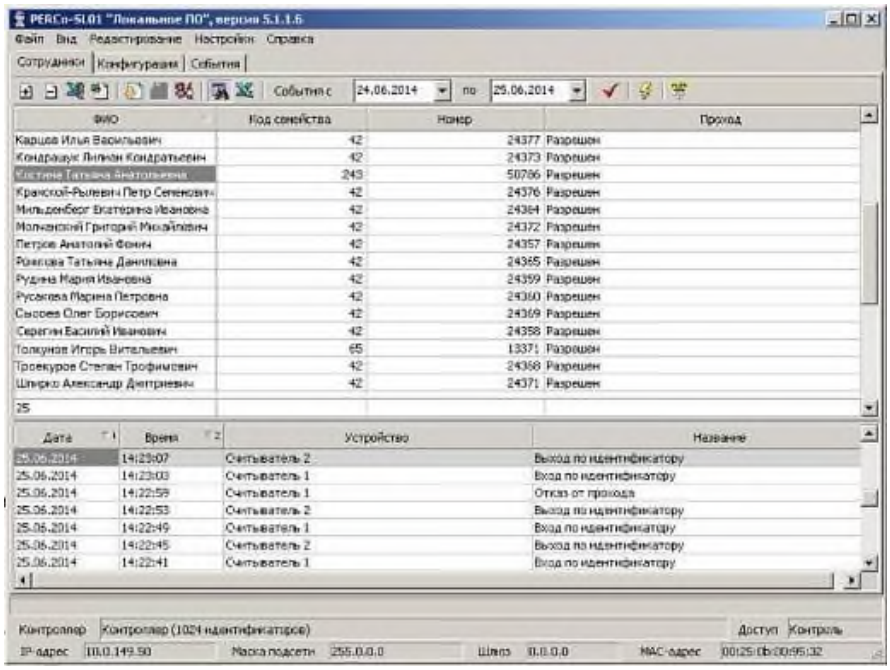

Рис. 3.1. Безкоштовне програмне забезпечення *PERCo – SL* 01

Так само існує *PERCo – SL* 02 «Локальне ПЗ з відеоідентифікації», яке дозволяє організувати захист від передачі перепустки іншій особі. На моніторі охоронця відображається фото власника карти, пред'явленої зчитувача. Охоронець має можливість порівняти фото з бази даних системи контролю доступу та особу пред'явника безконтактної карти або його зображення, якщо встановлена відеокамера.

Установка мережевого ПО комплексної системи безпеки *PERCo – S* -20 дозволяє на базі електронної прохідний вирішувати завдання безпеки і підвищення ефективності роботи підприємства. Крім контролю доступу можна організувати контроль порушень трудової дисципліни, автоматизація обліку робочого часу і розрахунку заробітної плати.

КТ02.3 може застосовуватися спільно з картоприймачем безконтактних карт доступу, дозволяючи організувати вилучення карт відвідувачів.

У моделі *КТ* 02.3 до вбудованого в стійку турнікета контролера можна підключити до 8-ми контролерів СКУД *PERCo – CL* 201 з вбудованим зчитувачем, що забезпечує економічне устаткування 8-ми приміщень системою контролю доступу.

Для формування зони проходу *КТ* 02.3 може бути доповнена секціями огорожі.

Вбудований замок механічного розблокування дозволяє відкрити турнікет за допомогою ключа, забезпечивши вільне обертання планок в обох напрямках.

3.2. Архітектура програмного забезпечення інтеграції з СКУД

Архітектура програмного забезпечення СКУД будується на основі ряду механізмів, які визначаються вимогами, що пред'являються до системи. Трактування СКУД як системи реального часу вимагає реалізації механізмів диспетчеризації, міжоб'єктної взаємодії і засобів роботи з таймерами. Паралелізм використовується для одночасної реалізаціюї зовнішніх подій і повинен забезпечуватись за рахунок використання багатопоточності. Клієнт-серверний підхід вносить необхідність реалізації механізму і способів взаємодії між сервером і додатками, а загальні вимоги безпеки і надійності змушують вибирати особливі способи зберігання даних і роботи з ними. Деякі архітектурні механізми використовують нестандартний підхід до створення об'єктів, що вимагає передачі строкового імені класу створюваного об'єкта. Цей підхід також може бути розглянуто як допоміжний механізм (або механізм нижчого рівня).

Диспетчер повідомлень. Для реалізації механізму межоб'єктної взаємодії був створений спеціальний об'єкт–диспетчер. Диспетчер «знає» всі об'єкти, які бажають обмінюватися повідомленнями, а ці об'єкти в свою чергу «знають» про існування диспетчера. Крім таблиці зареєстрованих об'єктів, важливою частиною диспетчера є черга повідомлень. Принцип відправки повідомлення можна описати таким чином:

– об'єкт-відправник створює повідомлення і ініціалізує його даними. Потім він викликає функцію відправки повідомлення (*SendEvent*) диспетчера;

– диспетчер здійснює постановку повідомлення в чергу, перевіряє, чи запущений потік обробки черги повідомлень. Якщо потік не запущений, то диспетчер його запускає. Після цього управління повертається об'єкту відправнику;

– потік обробки повідомлень витягує наступне повідомлення з черги, шукає об'єкт-одержувач по таблиці зареєстрованих об'єктів і викликає функцію

обробки повідомлення одержувача (*ProcEvent*), передаючи їй як параметр об’єкт повідомлення. Коли управління повертається диспетчеру, потік здійснює перевірку наявності повідомлень в черзі. Якщо повідомлення відсутні, то потік завершується, в іншому випадку потік здійснює обробку повідомлення.

Перед викликом функції *ProcEvent* одержувача здійснюється запуск таймера. Якщо до моменту таймаута управління не повернуто об’єкту диспетчера, то потік переривається і запускається знову з наступного елемента черги повідомлень. Якщо ж одержувач знає, що час обробки повідомлення перевищує час таймаута, то він запускає свій потік і повертає управління диспетчеру.

Черга повідомлень диспетчера є масивом елементів повідомлень. Повідомлення в чергу укладаються послідовно. При досягненні кінця черги наступне повідомлення записується на перше місце, таким чином, чергу зациклена. Обробка черги потоку припиняється, коли номер наступного оброблюваного елемента дорівнює номеру наступного елементу, що додається, а *це* означає, що в черзі відсутні повідомлення. Виділення об’єкта диспетчера з його чергою повідомлень вирішує всі проблеми синхронізації многопоточного додатку, за умови, що взаємодія об’єктів здійснюється тільки через чергу обробки повідомлень диспетчера. Кожен взаємодіючий об’єкт в системі характеризується трьома значеннями: номером класу, номером об’єкта і додатковим кодом. Для унікальної ідентифікації об’єкта використовуються перші два значення. Номери об’єктів видаються диспетчером послідовно, із заповненням пустот (тобто об’єкт займає перший вільний номер). Додатковий код покликаний виражати призначену для користувача нумерацію об’єктів. Крім того, він може бути використаний для пошуку об’єктів в системі (*Seek, Select*).

Трудомісткість пошуку елемента по таблиці взаємодіючих об’єктів є логарифмічною від числа об’єктів, що належать класу шуканого. Це пояснюється тим, що диспетчер працює з класами, як елементами масиву фіксованої довжини, а з об’єктами класу як з розширюваним масивом. Пошук по списку об’єктів здійснюється дихотомічно.

Проведене тестування показало, що диспетчер здатний підтримувати необмежену кількість об’єктів. Єдиним «вузьким» місцем може виступити черга повідомлень: оскільки вона зациклена, то її переповнення може закінчитися втратою ряду початкових повідомлень.

Диспетчер подій. Для реалізації можливості підписки одних об’єктів на події інших диспетчер повідомлень був розширений функціями диспетчера подій. Це виразилося в додаванні таблиці передплатників і функцій роботи з подіями.

Об’єкт, який бажає заявити про подію, створює об’єкт-повідомлення і заповнює частину його полів, що стосуються події. Після цього здійснюється виклик функції повідомлення про подію (*ThrowEvent*). Ця функція знаходить всіх передплатників, і відправляє їм повідомлення про подію, видаляючи їх підписні записи з таблиці. Якщо об’єкт бажає отримати подія знову, він повинен знову на нього підписатися.

Об’єкти мають можливість підписуватися на події монопольно (для цього в об’єкті повідомлення передбачений відповідний атрибут). При виникненні події диспетчер визначає наявність монопольних підписок, і розсилає повідомлення, повідомляючи передплатників, чи була заявлена монопольна підписка і чи є він монопольним передплатником. Залежно від цього об’єкт приймає рішення про відповідну реакцію на подію.

Диспетчер таймерів. Для зниження завантаженості системи таймерами було вирішено організувати службу таймерів на основі диспетчера повідомлень. Для цього були додані таблиці таймерів об’єктів. При запуску диспетчера автоматично запускається періодичний таймер. Після закінчення кожного періоду лічильник кожного об’єкта, який замовив таймер (*SetTimer*), зменшується на одиницю. При досягненні нульового значення об’єкту надсилається повідомлення про закінчення часу очікування, при цьому запис таймера не видаляється і може бути ініціалізоване знову повторним викликом *SetTimer*. Для видалення запису таймера використовується функція *DeleteTimer*.

Якщо викликається деструктор об’єкта, зареєстрованого у диспетчера, то відбувається видалення реєстрації даного об’єкта, видалення всіх його підписок

(як вхідних, так і вихідних), а також видалення всіх його таймерів. Крім цього організовується перегляд черги повідомлень диспетчера і видалення всіх повідомлень, спрямованих видаляємому об'єкту.

Трудомісткість пошуку по таблиці подій і таблиці таймерів логарифмічна від числа всіх елементів відповідної таблиці.

Клас *CEventDispatcher* реалізує диспетчер подій, повідомлень і службу таймерів. Клас *CObjEvent* є об'єктом подія, що містить інформацію про відправника, одержувача, тип повідомлення, структурі події і часу відправлення. Клас *CInteractionObject* задає загальний інтерфейс для всіх взаємодіючих об'єктів в системі. Структури *ITEM _ INTER _ OBJ, TIMER _ INFO, SUBSCRIBER _ INFO* представляють, відповідно, записи в таблиці об'єктів, таблиці таймерів і таблиці передплатників. Даний механізм реалізований в модулях *InteracitonObject*. *cpp* (*h*), *EventDispatcher*. *cpp* (*h*), *ObjEvent*. *cpp* (*h*).

3.3. Збереження об'єктів

Збереження об'єкта в файл. Деякі об'єкти відео системи вимагають збереження на диск і подальшого відновлення. Для цього застосовується механізм серіалізациі (перетворення об'єкта в послідовну форму і назад). Принцип роботи цього механізму наступний:

– при виконанні функції збереження в файл (*Save*) об'єкт насамперед записує рядок з ім'ям свого класу. Після цього відбувається виклик функції сериализации (*Serialize*) з параметром «зберегти»;

– у цій функції об'єкт здійснює запис на диск всіх своїх атрибутів. Після цього об'єкт вважається збереженим;

– при виконанні функції проштовхування об'єктів (*Load*) з файлу зчитується ім'я класу завантажується об'єкта. Цей рядок передається механізму параметризовано створених об'єктів, який здійснює створення відповідного об'єкта. Після чого викликається функція *Serialize* з параметром «завантажити».

У функції *Serialize* відбувається зчитування з файлу атрибутів об'єкту, що зберігається. Таким чином, ми можемо зберігати в одному файлі довільне число

об’єктів. Однак, щоб отримати доступ до будь-якого об’єкта, нам доведеться послідовно завантажити всі попередні об’єкти. Тому серіалізацію зручно використовувати лише для збереження стану одиночних об’єктів, або груп об’єктів, які використовуються одночасно. На рис. 3.2 представлена діаграма класів, що відображає структуру даного механізму.

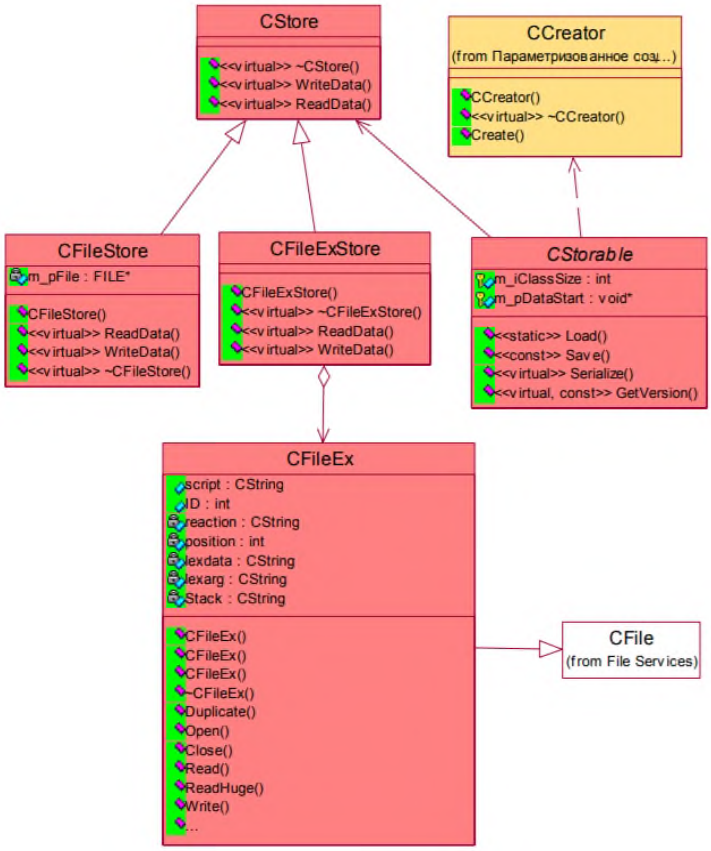

Рис. 3.2. Діаграма класів механізму серіализації

Для використання цього механізму всі класи, які бажають, щоб їх об’єкти зберігалися, повинні наслідувати від класу, що задає інтерфейс для серіалізації (*CStorable*, модуль *Storable*. *Cpp* (*h*)). Клас *CStore* задає інтерфейс для використання різних типів файлових сховищ. Наприклад, клас *CFileStore* використовує для роботи з файлами функції *API,* а клас *CFileExStore* – функції класу *CFileEx* (реалізація в модулі З *Store*. *H*). Крім них можуть бути використані будь-які класи-обгортки, що забезпечують роботу з файлами.

Цей механізм є повністю об’єктно-орієнтованим, так як тільки сам об’єкт знає порядок завантаження / збереження своїх атрибутів. Серіалізация реалізована і використовується в стандартній бібліотеці класів *MFC*.

Однак *MFC* -реалізація цього механізму може бути неприйнятна в випадках, коли застосовується множинне успадкування або передбачається забезпечити кросплатформеність додатків. В [11] описана альтернативна реалізація цього підходу, без залучення *MFC*. Вона вимагає використання механізму параметризрвані створення об'єктів, який буде описаний окремо. Крім цього, потрібно, щоб компілятор умів визначати інформацію про клас об'єкту під час виконання. Щоб прискорити процес запису об'єкта в файл, було прийнято наступне рішення. Замість того щоб записувати на диск по-черзі всі свої атрибути, об'єкт записує себе цілком шляхом копіювання області пам'яті.

Цей спосіб дозволяє заощадити на часі записи, але при цьому копіюються не тільки атрибути, а й таблиці віртуальних функцій об'єкта і всіх його базових класів. Однак таблиці віртуальних функцій об'єкта змінюються лише від компіляції до компіляції. Даний підхід може застосовуватися лише у випадках, коли необхідно проводити постійні збереження об'єктів.

Збереження в файл не дозволяє нам організувати швидкий пошук та вилучення об'єктів. Для вирішення цього завдання був розроблений спеціалізований клас *CStock*, що є контейнером одно- або двуключевих записів і реалізує можливості збереження і пошуку об'єктів (модуль С *Stock*. *Cpp* (*h*)).

Склад може зберігати тільки об'єкти одного типу. Для цього він ініціалізується ім'ям класу збережених об'єктів, положенням і довжиною ключів в запису, що характеризує об'єкт. Для розміщення об'єктів на склад використовується механізм, схожий на серіалізацію. Кожен об'єкт, який бажає зберігатися на складі, повинен мати функції, що задаються інтерфейсом *CStockItem* (модуль *CStockItem*. *Cpp* (*h*)).

При додаванні об'єкта на склад, викликається функція перетворення (*Transform*), яка за аналогією з функцією *Serialize* здійснює запис атрибутів об'єкта та їх вилучення. Однак в якості сховища атрибутів тут виступає не файл, а рядок (область пам'яті). Отриманий рядок додається в таблицю елементів і включається в індекс відповідно до свого ключем (ключами).

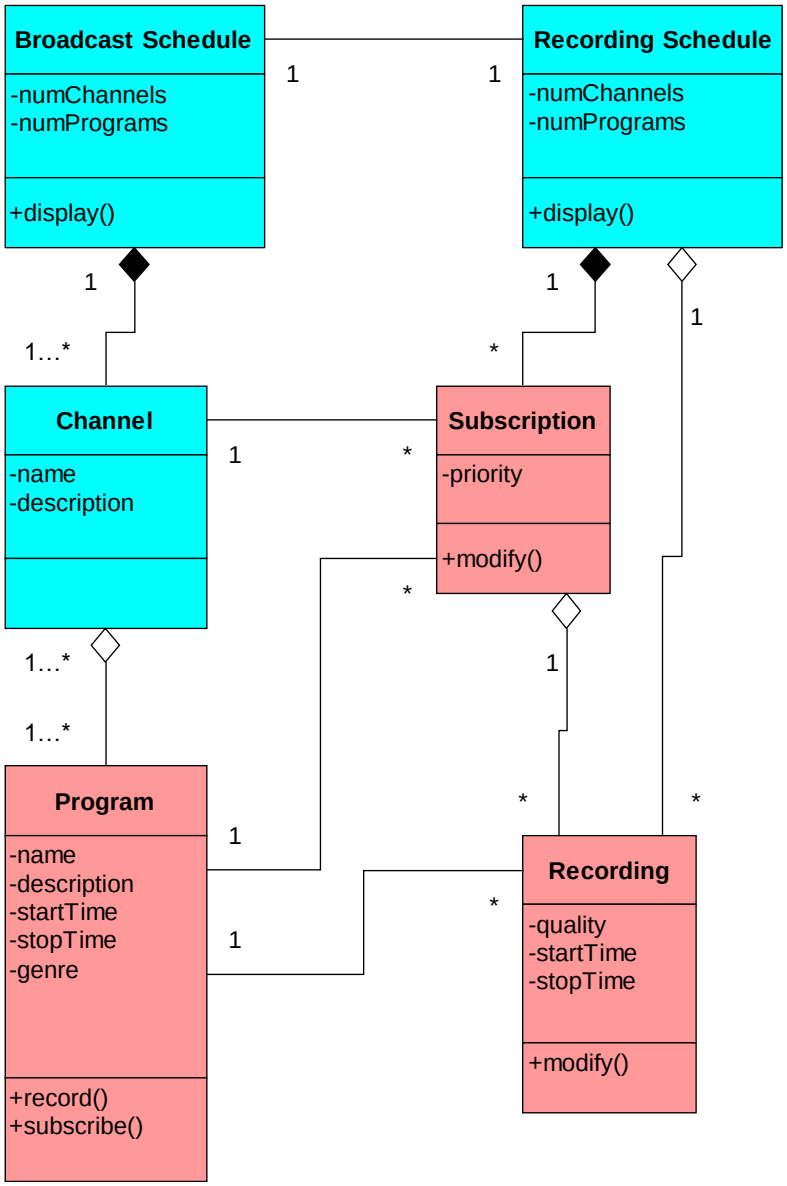

Рис. 3.3. Діаграма класів механізму складування об’єктів

Клас *CStock* має два індекси (по одному на кожен ключ), які використовуються для швидкого пошуку елементів. Пошук елементів по індексам здійснюється за допомогою дихотомії, з використанням лексикографічного порівняння, що дає можливість використання в якості ключів будь-яких типів, в тому числі і рядків. Зі складу можна витягувати або один елемент, що характеризується двома ключами, або всі елементи, що мають однакові значення якого-небудь одного ключа.

Реалізовано можливості вилучення елемента зі складу з блокуванням записи і без блокування. Для створення об’єктів видобутих елементів використовується механізм параметризації конструювання об’єктів.

3.4. Параметризоване створення об’єктів

Для параметризації створюваного об’єкта ім’ям класу використовувався механізм, описаний в шаблоні проектування «Абстрактна фабрика». Застосовуваний підхід також описаний в [11].

Клас *CAbstractFactory* задає загальний інтерфейс для створення продукту (об’єкта) різними фабриками об’єктів. Клас *CFactory* ініціалізується створюваним продуктом і реалізує інтерфейс абстрактної фабрики. Клас *CFactoryListItem* є елементом списку фабрик. Клас *CCreator* здійснює перебір списку фабрик для створення конкретного продукту, визначеного переданим йому параметром. Діаграма класів представлена на рис. 3.5.

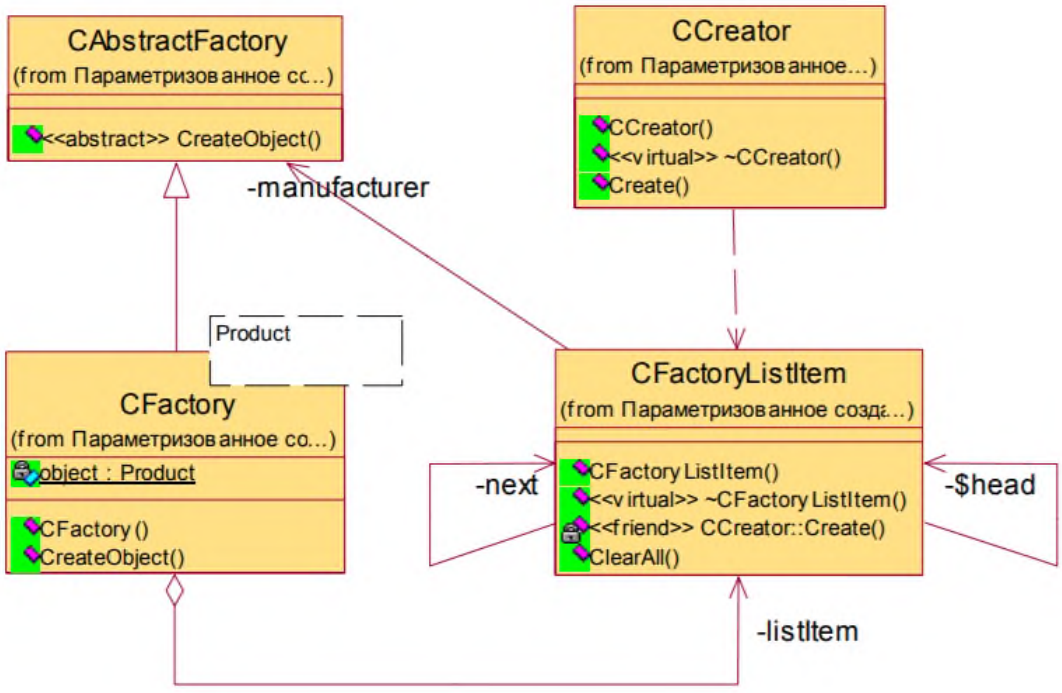

Рис. 3.5. Діаграма класів механізму параметризрвані створення об’єктів

Всі описані класи реалізовані в модулі *dynamic. cpp* (*h*) Загальна послідовність дій виглядає наступним чином:

– створюється новий об’єкт класу *CFactory*, параметризовані класом об’єкта продукту. При цьому автоматично здійснюється створення нового елемента *CFactoryListItem* і його додавання до існуючого списку фабрик;

– викликається функція *Create* класу *CCreator*, якій, як параметр, передається рядок, що містить ім’я класу створюваного об’єкта. Клас *CCreator* здійснює перегляд списку фабрик і викликає функцію створення продукту для кожної фабрики;

– фабрика перевіряє, чи співпадає переданий параметр з ім’ям класу її продукції, і в залежності від цього створює чи ні об’єкт. Для створення об’єкта використовується конструктор за замовчуванням;

– як тільки чергова фабрика повертає непорожнє значення, клас *CCreator* перериває перегляд списку фабрик і, в свою чергу, повертає створений об’єкт. Якщо жодна фабрика не змогла створити запитуваний об’єкт, то повертається порожній покажчик.

Для коректної роботи з продуктами, які використовують множинне спадкування, необхідно здійснити динамічне перетворення створеного об’єкта до його типу. Об’єкти-фабрики можна створювати один раз при старті програми і знищувати при її завершенні. З іншого боку, якщо параметризрвані створення об’єктів проводиться рідко, то фабрики можна створювати і видаляти по мірі необхідності.

3.5. Взаємодія додатків

Команди. Архітектура "клієнт-сервер» має на увазі, що програми-клієнти посилають запити серверу, а той їх виконує. Часто до сервера висуваються вимоги організації протоколювання цих запитів. Для реалізації такої схеми був використаний шаблон проектування «Команда». Він інкапсулює запит як об’єкт, дозволяючи тим самим задавати параметри клієнтів для обробки відповідних запитів, ставити запити в чергу або протоколювати їх, а також підтримувати скасування операцій.

Всі команди мають загальний інтерфейс, що задається класом *CCommand*. В цьому класі визначена віртуальна функція *Execute*, яка ініціює виконання команди. Такий підхід дозволяє виконати команду, не знаючи, яка це команда і що вона повинна зробити. Якщо потрібно виконати не одну команду, а кілька, то

можна визначити клас *CMacroCommand* як спадкоємець класу *CCommand*, доповнивши його операціями додавання і видалення команди і переписавши операцію *Execute* таким чином, щоб вона здійснювала послідовне виконання списку команд.

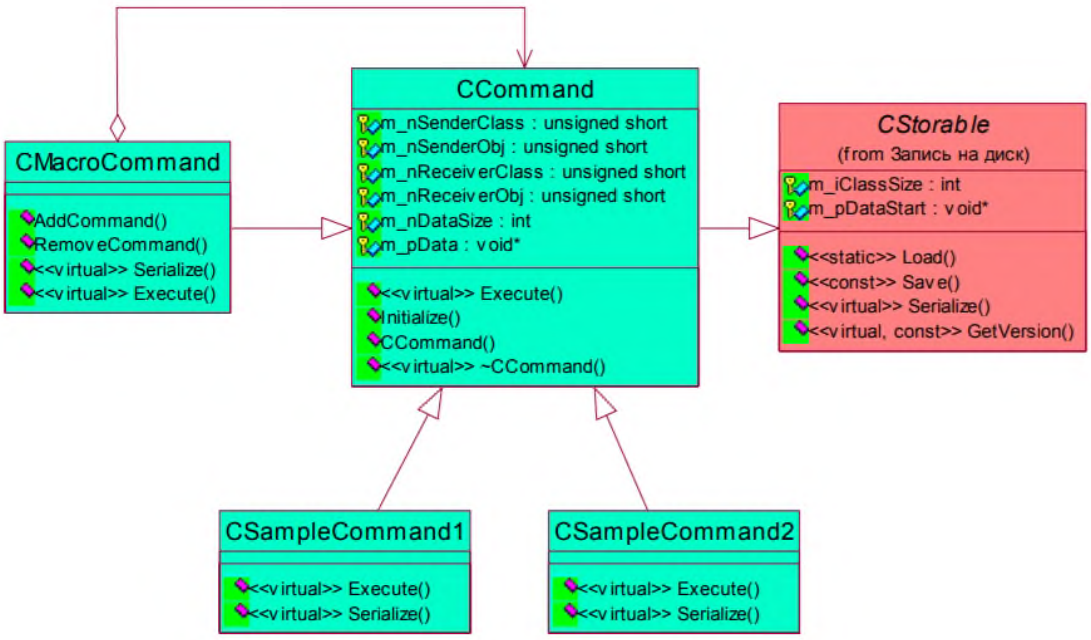

Рис. 3.6. Діаграма класів механізму команд

Для того щоб організувати протоколювання команд скористаємося механізмом збереження об’єктів в файл, описаним вище. Для цього успадкуємо клас *CCommand* (модуль *Command. H*) від класу *CStorable*. Це дозволить здійснити послідовну запис команд у файл. Протокол може бути використаний для відновлення стану системи в разі збою. Передача об’єктів. Сервер і клієнти проводять обмін даними. Часто ці дані представляють собою різні об’єкти. Для того щоб організувати передачу об’єкта від однієї програми іншому, можна скористатися механізмом, аналогічним серіалізації, і здійснювати перетворення об’єкта в рядок. Тоді весь процес передачі об’єкта буде виглядати наступним чином:

– додаток-відправник створює об’єкт, ініціалізує його даними і викликає функцію перетворення в рядок (*Transform*);

– отриманий рядок передається з додатком одержувачу разом з ім’ям класу об’єкта (або іншим ідентифікатором, що визначає об’єкт).

– одержувач створює об’єкт і викликає функцію перетворення, передаючи їй отриманий рядок. Використання цього механізму передбачає, що і відправник і одержувач знають про клас об’єктів, що передаються. Істотною перевагою же є той факт, що тільки самі об’єкти знають, як себе упакувати. Це дозволяє не розробляти додатковий формат пакета для передачі кожного об’єкта і полегшує можливості заміни об’єктів і їх повторного використання, оскільки змінити потрібно лише логіку функцій класу об’єкта.

Для того щоб мати можливість уніфікованого перетворення в рядок був розроблений клас *CTransformable* (модуль *Transformable*. *H*). Цей клас також був використаний в механізмі збереження об’єктів на склад.

3.5.1. Пакет «Система зв’язку сервера»

Розглянемо склад і принципи роботи системи зв’язку додатку з сервером в СКУД (рис. 3.7).

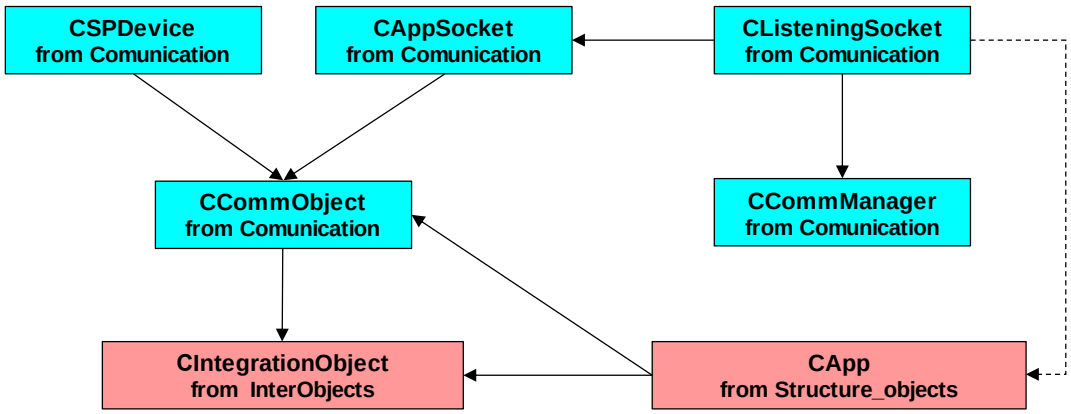

Рис. 3.7. Пакет «Система зв’язку сервера»

Клас *CCommManager* задає загальний інтерфейс для прийому вхідних з’єднань. При прийомі з’єднання створюється об’єкт класу *CCommObject*, через який і здійснюється подальша взаємодія з під’єднаним додатком. В даному випадку в якості комунікаційних об’єктів були обрані сокети (для їх об’єктно-орієнтованого уявлення використовується клас *CBlockingSocket*, що є обгорткою функцій *API Winsock*). Однак в разі необхідності сокети можуть бути замінені будь-яким іншим засобом міжпроцесної взаємодії (каналами, файловими

чергами і ін.). Крім комунікаційного об’єкта створюється структурний об’єкт, який представляє собою образ додатку в системі (*CApp*). Образ додатки знає реквізити свого комунікаційного об’єкта і може спілкуватися з ним через диспетчер, що відображено на діаграмі стрілкою з позначкою « *disp _ link* ». Зв’язок з контролерами забезпечується за допомогою плати зв’язку, підключеної до комп’ютера. Клас *CSPDevice* представляє в системі драйвер плати зв’язку. Він теж успадковує інтерфейс комунікаційного об’єкта. Такий підхід дозволяє уніфікувати обробку зовнішніх сигналів в системі.

Кожен пакет даних, отриманий комунікаційним об’єктом, вважається системною подією. Наприклад, подія «вставлена карта» від контролера або подія «новий клієнт» від програми. Дані про об’єкт-джерелі і тип події закладені в формат пакета. Використовуючи ці відомості, комунікаційний об’єкт здійснює публікацію події від імені джерела.

Комунікаційний об’єкт здійснює відправку тільки готових пакетів, упаковкою же даних в пакет з боку сервера займається образ додатки *CApp*.

3.5.2. Структурні об’єкти

Всі структурні елементи системи контролю доступу мають спеціальних агентів, які представляють інтереси даних елементів всередині програми-сервера. Взаємодія між агентами та іншими об’єктами сервера відбувається за допомогою диспетчера (інтерфейс *CInteractionObject*). Для забезпечення надійності системи структурні об’єкти зберігаються в файл (інтерфейс *CStorable*). У разі необхідності можна передати об’єкт з усіма його даними іншому додатку у вигляді рядка (інтерфейс *CTransformable*).

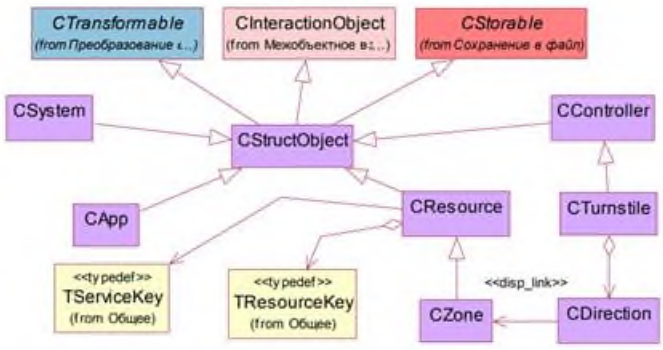

Рис. 3.8. Пакет «Структурні об’єкти»

На рис. 3.8. представлені класи пакету «Структурні об’єкти». Клас *CSystem* являє собою систему і є джерелом системних подій. Клас *CApp* є джерелом для об’єктів-агентів додатків, підключених до сервера. Клас *CController* задає загальний інтерфейс для різних типів контролерів, а *CResource* для різних ресурсів. Таким чином, наслідуючи від цих класів, ми отримуємо можливість створення не тільки СКУД, а й деяких інших систем автоматизації маркетингу.

Для ресурсів-приміщень в якості контролера повинен використовуватися клас турнікета (*CTurnstile*), а клас *CZone* в якості ресурсу. Для конкретного класу *CTurnstile* важливий напрямок доступу: з напрямком доступу пов’язана відповідна зона-ресурс. Усі матеріали в системі послідовно пронумеровані для організації можливості обмеження доступу конкретного користувача до вибраного ресурсу (*TResourceKey*). Ключі використовуються для організації пошуку пов’язаних елементів на складах.

3.6. Висновки до розділу

Розроблена система інтеграції в систему безпеки відповідає сучасним тенденціям, але не варто забувати і про перспективи розвитку інтегрованих систем безпеки. Основні напрями визначаються наступними вимогами:

– зниження ролі людини в процесі забезпечення безпеки за рахунок підвищення інтелектуальності систем;

– зниження рівня помилкових спрацьовувань за рахунок більш тісної використання підсистем;

– вимога відкритості. Розробники ІСБ повинні забезпечити замовнику за допомогою відкритих протоколів можливості підключення систем і устаткування інших виробників і гнучкого настроювання ІСБ під свої потреби.

Реалізація зазначених вимог з одного боку дозволить збільшити ефективність систем безпеки, зменшить вплив людського фактору, з іншого - зробить побудова інтегрованих систем більш прозорим.

ВИСНОВКИ

У дипломному проєкті було розроблено програмні модулі для інтеграції відеоспостереження в існуючу СКУД.

Була розроблена схема для представлення архітектури додатку-сервера СКУД:

– «Система зв’язку сервера» – пакет, який відповідає за зв’язок клієнтів і серверу, а також серверу і устаткування СКУД, такого як турнікети, датчики і т.д. Він організовує прийом вхідних повідомлень і відправку вихідних. Вхідні повідомлення інтерпретуються як події, їх публікація здійснюється від імені структурних об’єктів;

– пакет «Структурні об’єкти» містить об’єкти, які є агентами зовнішніх елементів системи: контролерів, програм-клієнтів та інших. Агенти використовують систему зв’язку для спілкування з зовнішнім світом;

– пакет «Об’єкти-дані» здійснює угруповання ряду пакетів, що представляють такі елементи предметної області, як карти, рахунки, групи, ресурси і персони;

– пакет «Виконавча підсистема» відповідає за виконання запитів від додатків і обробку подій контролерів. Він являє собою адаптовану до системи реалізацію механізму команд. Команди працюють як з об’єктами-даними, так і зі структурними об’єктами;

– пакет «Збереження об’єктів» показує, яким чином сервер використовує механізми збереження об’єктів. Цей пакет використовується командами, структурними об’єктами і об’єктами-даними.

СПИСОК БІБЛІОГРАФІЧНИХ ПОСИЛАНЬ ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Абалмазов Э.И. Энциклопедия безопасности : справочник. М.: КНОРУС, 2011. 516с.

2. Андрусенко С.И. Интегрированные системы безопасности на крупных объектах // Все о вашей безопасности. 2011. № 2. С.67-73.

3. Абрамов А.М. Системы управления доступом. М.: "Оберег-РБ", 2013. 238с.

4. Барсуков В.С. Безопасность: технологии, средства, услуги: учеб.пособие. М.: Юнити, 2011. 280с.

5. Барсуков В.С. Интегральная защита информации // Системы безопасности, 2012. №5. С.102-103.

6. Гензберк Ю.Н. Охранное телевидение : учеб.пособие. М.: Пресс-Медиа, 2011. 312с.

7. Генне О.В. Интеграции систем видеонаблюдения // Защита информации. М.: Конфидент, 2012. №1. С.28-31.

8. Груба И.И. Системы охранной сигнализации : учеб.пособие. М.: СОЛОН-ПРЕСС, 2012. 220 с.

9. Крахмалев А.К. Интегрированная система безопасности: учеб.пособие. М.: Москва, 2013. 243с.

10. Крахмалев А.К. Средства и системы контроля и управления доступом : учеб.пособие. М.: НИЦ "Охрана" ГУВО МВД России. 2013.

11. Мялянов В.С. Мир связи : учеб.пособие. Спб.: Наука и техника, 2012. 214*с*.

12. Омельянчук А.М. Интегрированные системы безопасности //Защита информации. 2012. № 2. С.49-51.

13. Омельянчук А.М. *Integratiosapiens* // Все о вашей безопасности. 2011. № 2. С.78-84.

14. Рыжов В.А. Проектирование и исследование комплексных систем безопасности : учеб.пособие. Спб.: НИУ ИТМО, 2013. 156 с.

15. Сабынин В.Н. Организация пропускного режима первый шаг к обеспечению безопасности и конфиденциальности информации // Информост радиоэлектроники и телекоммуникации, 2011. №3 . С.32-35.

16. Тарасов Ю.А. Контрольно-пропускной режим на предприятии. Защита информации // Конфидент, 2012. № 1. С.54-57.

17. Хализев В.Н. Математическая модель синтеза интегрированной системы безопасности // Научный журнал КубГАУ, 2012. №81(07) С.1-15.

18. ДСТУ 3008-95 Структура і правила оформлення. Документація. Звіти у сфері науки і техніки

19. Бойченко С.В., Іванченко О.В. Положення про дипломні роботи (проекти) випускників Національного авіаційного університету. – К.: НАУ, 2017. – 63 с.

Додаток А

Лістинг основного програмного модуля

```
import wx
import socket
import threading
import time
import  wx.lib.masked          as  masked
import  wx.lib.scrolledpanel     as  scroll
class FuncThread(threading.Thread):
    def __init__(self, target, *args):
        self._target = target
        self._args = args
        threading.Thread.__init__(self)
    def run(self):
        self._target(*self._args)
def ByteToHex(byteStr):
    return ‘‘.join([ "%02x " % ord(x) for x in byteStr ]).strip()
class TestTool(wx.Frame):
    def __init__(self, *args, **kwds):
        # begin wxGlade: MyFrame.__init__
        kwds["style"] = wx.DEFAULT_FRAME_STYLE
        wx.Frame.__init__(self, *args, **kwds)
        self.SetFont(wx.Font(10, wx.DEFAULT, wx.NORMAL, wx.NORMAL, 0,
""))
        self.l_ip = wx.StaticText(self, -1, "IP")
        self.tc_ip = masked.Ctrl(self, -1,controlType =
masked.controlTypes.IPADDR,style=wx.TE_CENTRE)
        self.l_port = wx.StaticText(self, -1, "Port")
```

```
        self.tc_port = masked.NumCtrl (self, -1,integerWidth = 5,groupDigits =
False,style=wx.TE_CENTRE)
        self.tc_port.SetBounds(0,65535)
        self.button_send = wx.Button(self, -1, u"Начать тестирование")
        #self.button_reset = wx.Button(self, -1, u"Сброс")
        self.l_speed_header = wx.StaticText(self, -1, u"Скорость подключения")
        self.l_open_header = wx.StaticText(self, -1, u"Открытие")
        self.l_send_header = wx.StaticText(self, -1, u"Запись")
        self.l_send_control_header = wx.StaticText(self, -1, u"Запись[проверка]")
        self.l_close_header = wx.StaticText(self, -1, u"Закрытие")
        self.l_speed110 = wx.StaticText(self, -1, u"110 бод")
        self.l_open_speed110 = wx.StaticText(self, -1, u"...")
        self.l_send_speed110 = wx.StaticText(self, -1, u"...")
        self.l_send_control_speed110 = wx.StaticText(self, -1, u"...")
        self.l_close_speed110 = wx.StaticText(self, -1, u"...")
        self.l_speed300 = wx.StaticText(self, -1, u"300 бод")
        self.l_open_speed300 = wx.StaticText(self, -1, u"...")
        self.l_send_speed300 = wx.StaticText(self, -1, u"...")
        self.l_send_control_speed300 = wx.StaticText(self, -1, u"...")
        self.l_close_speed300 = wx.StaticText(self, -1, u"...")
        self.l_speed600 = wx.StaticText(self, -1, u"600 бод")
        self.l_open_speed600 = wx.StaticText(self, -1, u"...")
        self.l_send_speed600 = wx.StaticText(self, -1, u"...")
        self.l_send_control_speed600 = wx.StaticText(self, -1, u"...")
        self.l_close_speed600 = wx.StaticText(self, -1, u"...")
        self.l_speed1200 = wx.StaticText(self, -1, u"1200 бод")
        self.l_open_speed1200 = wx.StaticText(self, -1, u"...")
        self.l_send_speed1200 = wx.StaticText(self, -1, u"...")
        self.l_send_control_speed1200 = wx.StaticText(self, -1, u"...")
        self.l_close_speed1200 = wx.StaticText(self, -1, u"...")
        self.l_speed2400 = wx.StaticText(self, -1, u"2400 бод")
```

```
self.l_open_speed2400 = wx.StaticText(self, -1, u"...")
self.l_send_speed2400 = wx.StaticText(self, -1, u"...")
self.l_send_control_speed2400 = wx.StaticText(self, -1, u"...")
self.l_close_speed2400 = wx.StaticText(self, -1, u"...")
self.l_speed4800 = wx.StaticText(self, -1, u"4800 бод")
self.l_open_speed4800 = wx.StaticText(self, -1, u"...")
self.l_send_speed4800 = wx.StaticText(self, -1, u"...")
self.l_send_control_speed4800 = wx.StaticText(self, -1, u"...")
self.l_close_speed4800 = wx.StaticText(self, -1, u"...")
self.l_speed9600 = wx.StaticText(self, -1, u"9600 бод")
self.l_open_speed9600 = wx.StaticText(self, -1, u"...")
self.l_send_speed9600 = wx.StaticText(self, -1, u"...")
self.l_send_control_speed9600 = wx.StaticText(self, -1, u"...")
self.l_close_speed9600 = wx.StaticText(self, -1, u"...")
self.l_speed19200 = wx.StaticText(self, -1, u"19200 бод")
self.l_open_speed19200 = wx.StaticText(self, -1, u"...")
self.l_send_speed19200 = wx.StaticText(self, -1, u"...")
self.l_send_control_speed19200 = wx.StaticText(self, -1, u"...")
self.l_close_speed19200 = wx.StaticText(self, -1, u"...")
self.l_speed38400 = wx.StaticText(self, -1, u"38400 бод")
self.l_open_speed38400 = wx.StaticText(self, -1, u"...")
self.l_send_speed38400 = wx.StaticText(self, -1, u"...")
self.l_send_control_speed38400 = wx.StaticText(self, -1, u"...")
self.l_close_speed38400 = wx.StaticText(self, -1, u"...")
self.l_speed57600 = wx.StaticText(self, -1, u"57600 бод")
self.l_open_speed57600 = wx.StaticText(self, -1, u"...")
self.l_send_speed57600 = wx.StaticText(self, -1, u"...")
self.l_send_control_speed57600 = wx.StaticText(self, -1, u"...")
self.l_close_speed57600 = wx.StaticText(self, -1, u"...")
self.l_speed115200 = wx.StaticText(self, -1, u"115200 бод")
self.l_open_speed115200 = wx.StaticText(self, -1, u"...")
```

```
        self.l_send_speed115200 = wx.StaticText(self, -1, u"...")
        self.l_send_control_speed115200 = wx.StaticText(self, -1, u"...")
        self.l_close_speed115200 = wx.StaticText(self, -1, u"...")
        self.label_array = []
        self.label_array.append([u"Скорость 110", self.l_speed110,
self.l_open_speed110, self.l_send_speed110,
                                 self.l_send_control_speed110, self.l_close_speed110])
        self.label_array.append([u"Скорость 300", self.l_speed300,
self.l_open_speed300, self.l_send_speed300,
                                 self.l_send_control_speed300, self.l_close_speed300])
        self.label_array.append([u"Скорость 600", self.l_speed600,
self.l_open_speed600, self.l_send_speed600,
                                 self.l_send_control_speed600, self.l_close_speed600])
        self.label_array.append([u"Скорость 1200", self.l_speed1200,
self.l_open_speed1200, self.l_send_speed1200,
                                 self.l_send_control_speed1200, self.l_close_speed1200])
        self.label_array.append([u"Скорость 2400", self.l_speed2400,
self.l_open_speed2400, self.l_send_speed2400,
                                 self.l_send_control_speed2400, self.l_close_speed2400])
        self.label_array.append([u"Скорость 4800", self.l_speed4800,
self.l_open_speed4800, self.l_send_speed4800,
                                 self.l_send_control_speed4800, self.l_close_speed4800])
        self.label_array.append([u"Скорость 9600", self.l_speed9600,
self.l_open_speed9600, self.l_send_speed9600,
                                 self.l_send_control_speed9600, self.l_close_speed9600])
        self.label_array.append([u"Скорость 19200", self.l_speed19200,
self.l_open_speed19200, self.l_send_speed19200,
                                 self.l_send_control_speed19200,
self.l_close_speed19200])
        self.label_array.append([u"Скорость 38400", self.l_speed38400,
self.l_open_speed38400, self.l_send_speed38400,
```

```
                    self.l_send_control_speed38400, self.l_close_speed38400])
        self.label_array.append([u"Скорость 57600",    self.l_speed57600,
self.l_open_speed57600,    self.l_send_speed57600,
                    self.l_send_control_speed57600,
self.l_close_speed57600])
        self.label_array.append([u"Скорость 115200",   self.l_speed115200,
self.l_open_speed115200,   self.l_send_speed115200,
                    self.l_send_control_speed115200,
self.l_close_speed115200])
        self.tcp_data_array = []
        self.tcp_data_array.append([u"110",    "\x81\x00\x00\x10"])
        self.tcp_data_array.append([u"300",    "\x81\x00\x00\x11"])
        self.tcp_data_array.append([u"600",    "\x81\x00\x00\x12"])
        self.tcp_data_array.append([u"1200",   "\x81\x00\x00\x13"])
        self.tcp_data_array.append([u"2400",   "\x81\x00\x00\x14"])
        self.tcp_data_array.append([u"4800",   "\x81\x00\x00\x15"])
        self.tcp_data_array.append([u"9600",   "\x81\x00\x00\x16"])
        self.tcp_data_array.append([u"19200",  "\x81\x00\x00\x17"])
        self.tcp_data_array.append([u"38400",  "\x81\x00\x00\x18"])
        self.tcp_data_array.append([u"57600",  "\x81\x00\x00\x19"])
        self.tcp_data_array.append([u"115200", "\x81\x00\x00\x1a"])
        self.sb = self.CreateStatusBar(1, 0)
        self.tc_ip.SetValue('192.168.10.10')
        self.tc_port.SetValue('8090')
        self.__set_properties()
        self.__do_layout()
        self.__attach_events()
        #self.demo()
    def __attach_events(self):
        self.Bind(wx.EVT_BUTTON, self.button_start, self.button_send)
        #self.Bind(wx.EVT_BUTTON, self.label_reset, self.button_reset)
```

```
            self.Bind(masked.EVT_NUM, self.port_correction, self.tc_port)
        def __set_properties(self):
            self.SetTitle("Test tool")
            self.SetMinSize((750, 400))
            self.SetSize((750,400))
            self.SetMaxSize((950, 650))
            self.sb.SetStatusWidths([-1])
            self.sb.SetStatusText(u"",0)

self.SetBackgroundColour(wx.SystemSettings_GetColour(wx.SYS_COLOUR_BTNFACE))
        def __do_layout(self):
            # begin wxGlade: MyFrame.__do_layout
            sizer_top = wx.BoxSizer(wx.VERTICAL)
            sizer_send = wx.BoxSizer(wx.HORIZONTAL)
            sizer_header = wx.BoxSizer(wx.HORIZONTAL)
            sizer_speed110 = wx.BoxSizer(wx.HORIZONTAL)
            sizer_speed300 = wx.BoxSizer(wx.HORIZONTAL)
            sizer_speed600 = wx.BoxSizer(wx.HORIZONTAL)
            sizer_speed1200 = wx.BoxSizer(wx.HORIZONTAL)
            sizer_speed2400 = wx.BoxSizer(wx.HORIZONTAL)
            sizer_speed4800 = wx.BoxSizer(wx.HORIZONTAL)
            sizer_speed9600 = wx.BoxSizer(wx.HORIZONTAL)
            sizer_speed19200 = wx.BoxSizer(wx.HORIZONTAL)
            sizer_speed38400 = wx.BoxSizer(wx.HORIZONTAL)
            sizer_speed57600 = wx.BoxSizer(wx.HORIZONTAL)
            sizer_speed115200 = wx.BoxSizer(wx.HORIZONTAL)
            sizer_send.Add(self.l_ip, 0, wx.ALL, 5)
            sizer_send.Add(self.tc_ip, 1, wx.ALL, 5)
            sizer_send.Add(self.l_port, 0, wx.ALL, 5)
            sizer_send.Add(self.tc_port, 1, wx.ALL, 5)
```

```
sizer_send.Add(self.button_send, 1, wx.ALL, 5)
#sizer_send.Add(self.button_reset, 1, wx.ALL, 5)
sizer_header.Add(self.l_speed_header, 2, wx.ALL, 5)
sizer_header.Add(self.l_open_header, 1, wx.ALL, 5)
sizer_header.Add(self.l_send_header, 1, wx.ALL, 5)
sizer_header.Add(self.l_send_control_header, 1, wx.ALL, 5)
sizer_header.Add(self.l_close_header, 1, wx.ALL, 5)
sizer_speed110.Add(self.l_speed110, 2, wx.ALL, 5)
sizer_speed110.Add(self.l_open_speed110, 1, wx.ALL, 5)
sizer_speed110.Add(self.l_send_speed110, 1, wx.ALL, 5)
sizer_speed110.Add(self.l_send_control_speed110, 1, wx.ALL, 5)
sizer_speed110.Add(self.l_close_speed110, 1, wx.ALL, 5)
sizer_speed300.Add(self.l_speed300, 2, wx.ALL, 5)
sizer_speed300.Add(self.l_open_speed300, 1, wx.ALL, 5)
sizer_speed300.Add(self.l_send_speed300, 1, wx.ALL, 5)
sizer_speed300.Add(self.l_send_control_speed300, 1, wx.ALL, 5)
sizer_speed300.Add(self.l_close_speed300, 1, wx.ALL, 5)
sizer_speed600.Add(self.l_speed600, 2, wx.ALL, 5)
sizer_speed600.Add(self.l_open_speed600, 1, wx.ALL, 5)
sizer_speed600.Add(self.l_send_speed600, 1, wx.ALL, 5)
sizer_speed600.Add(self.l_send_control_speed600, 1, wx.ALL, 5)
sizer_speed600.Add(self.l_close_speed600, 1, wx.ALL, 5)
sizer_speed1200.Add(self.l_speed1200, 2, wx.ALL, 5)
sizer_speed1200.Add(self.l_open_speed1200, 1, wx.ALL, 5)
sizer_speed1200.Add(self.l_send_speed1200, 1, wx.ALL, 5)
sizer_speed1200.Add(self.l_send_control_speed1200, 1, wx.ALL, 5)
sizer_speed1200.Add(self.l_close_speed1200, 1, wx.ALL, 5)
sizer_speed2400.Add(self.l_speed2400, 2, wx.ALL, 5)
sizer_speed2400.Add(self.l_open_speed2400, 1, wx.ALL, 5)
sizer_speed2400.Add(self.l_send_speed2400, 1, wx.ALL, 5)
sizer_speed2400.Add(self.l_send_control_speed2400, 1, wx.ALL, 5)
```

```
sizer_speed2400.Add(self.l_close_speed2400, 1, wx.ALL, 5)
sizer_speed4800.Add(self.l_speed4800, 2, wx.ALL, 5)
sizer_speed4800.Add(self.l_open_speed4800, 1, wx.ALL, 5)
sizer_speed4800.Add(self.l_send_speed4800, 1, wx.ALL, 5)
sizer_speed4800.Add(self.l_send_control_speed4800, 1, wx.ALL, 5)
sizer_speed4800.Add(self.l_close_speed4800, 1, wx.ALL, 5)
sizer_speed9600.Add(self.l_speed9600, 2, wx.ALL, 5)
sizer_speed9600.Add(self.l_open_speed9600, 1, wx.ALL, 5)
sizer_speed9600.Add(self.l_send_speed9600, 1, wx.ALL, 5)
sizer_speed9600.Add(self.l_send_control_speed9600, 1, wx.ALL, 5)
sizer_speed9600.Add(self.l_close_speed9600, 1, wx.ALL, 5)
sizer_speed19200.Add(self.l_speed19200, 2, wx.ALL, 5)
sizer_speed19200.Add(self.l_open_speed19200, 1, wx.ALL, 5)
sizer_speed19200.Add(self.l_send_speed19200, 1, wx.ALL, 5)
sizer_speed19200.Add(self.l_send_control_speed19200, 1, wx.ALL, 5)
sizer_speed19200.Add(self.l_close_speed19200, 1, wx.ALL, 5)
sizer_speed38400.Add(self.l_speed38400, 2, wx.ALL, 5)
sizer_speed38400.Add(self.l_open_speed38400, 1, wx.ALL, 5)
sizer_speed38400.Add(self.l_send_speed38400, 1, wx.ALL, 5)
sizer_speed38400.Add(self.l_send_control_speed38400, 1, wx.ALL, 5)
sizer_speed38400.Add(self.l_close_speed38400, 1, wx.ALL, 5)
sizer_speed57600.Add(self.l_speed57600, 2, wx.ALL, 5)
sizer_speed57600.Add(self.l_open_speed57600, 1, wx.ALL, 5)
sizer_speed57600.Add(self.l_send_speed57600, 1, wx.ALL, 5)
sizer_speed57600.Add(self.l_send_control_speed57600, 1, wx.ALL, 5)
sizer_speed57600.Add(self.l_close_speed57600, 1, wx.ALL, 5)
sizer_speed115200.Add(self.l_speed115200, 2, wx.ALL, 5)
sizer_speed115200.Add(self.l_open_speed115200, 1, wx.ALL, 5)
sizer_speed115200.Add(self.l_send_speed115200, 1, wx.ALL, 5)
sizer_speed115200.Add(self.l_send_control_speed115200, 1, wx.ALL, 5)
sizer_speed115200.Add(self.l_close_speed115200, 1, wx.ALL, 5)
```

```
        sizer_top.Add(sizer_send, 1, wx.EXPAND, 0)
        sizer_top.Add(sizer_header, 1, wx.EXPAND, 0)
        sizer_top.Add(sizer_speed110, 1, wx.EXPAND, 0)
        sizer_top.Add(sizer_speed300, 1, wx.EXPAND, 0)
        sizer_top.Add(sizer_speed600, 1, wx.EXPAND, 0)
        sizer_top.Add(sizer_speed1200, 1, wx.EXPAND, 0)
        sizer_top.Add(sizer_speed2400, 1, wx.EXPAND, 0)
        sizer_top.Add(sizer_speed4800, 1, wx.EXPAND, 0)
        sizer_top.Add(sizer_speed9600, 1, wx.EXPAND, 0)
        sizer_top.Add(sizer_speed19200, 1, wx.EXPAND, 0)
        sizer_top.Add(sizer_speed38400, 1, wx.EXPAND, 0)
        sizer_top.Add(sizer_speed57600, 1, wx.EXPAND, 0)
        sizer_top.Add(sizer_speed115200, 1, wx.EXPAND, 0)
        self.SetSizer(sizer_top)
        #sizer_top.Fit(self)
        self.Layout()
    def button_start(self,event):
        test_thread = FuncThread(self.start_test)
        test_thread.start()
    def start_test(self):
        self.label_reset(None)
        """
        i=0
        for data in self.tcp_data_array:
            print data[0]
            print ByteToHex(data[1])
            self.label_array[i][2].SetLabel("OK")
            i=i+1
        for data in self.tcp_data_array:
            print ByteToHex(data[1][2])
            print ByteToHex(data[1][3])
```

```
            """
            s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            TCP_IP = self.tc_ip.GetValue()
            TCP_PORT = int(self.tc_port.GetValue())
            s.settimeout(10)
            try:
               s.connect((TCP_IP, TCP_PORT))
            except socket.error:
               self.sb.SetStatusText(u"Подключение не удалось",0)
            else:
               s.settimeout(None)
               self.sb.SetStatusText(u"Успешное подключение. Идет
тестирование...",0)
               i=0
               for data in self.tcp_data_array:
                  s.send(data[1])
                  print data[0]
                  answer = s.recv(4)
                  if answer[1] == "\x01" and answer[2]== data[1][2] and
answer[3]== data[1][3]:
                     self.label_array[i][2].SetLabel("OK")
                  #write
                     s.send("\x02\x00\x00\x01\x55")
                     write_answer = s.recv(4)
                     if write_answer[1] == "\x01":
                           self.label_array[i][3].SetLabel("OK")
                  #write and check
                     s.send("\x02\x01\x00\x01\xAA")
                     write_answer = s.recv(4)
                     if write_answer[1] == "\x01":
                           self.label_array[i][3].SetLabel("OK")
```

```
                    elif write_answer[1] == "\x03":
                        self.label_array[i][3].SetLabel("Error")
                #close
                    s.send("\x82\x00\x00\x00")
                    close_answer = s.recv(4)
                    if close_answer[1] == "\x00" and answer[2]== data[1][2] and
answer[3]== data[1][3]:
                            self.label_array[i][5].SetLabel("OK")
                elif answer[1] == "\x01" and answer[3]!= data[1][3]:
                    for datax in self.tcp_data_array:
                        if answer[3] == datax[1][3]:
                            self.label_array[i][2].SetLabel(datax[0])
                elif answer[1] == "\x00":
                    for datax in self.tcp_data_array:
                        if answer[3] == datax[1][3]:
                            self.label_array[i][2].SetLabel("Can't open")
                i=i+1
                time.sleep(0.1)
            s.close()
            self.sb.SetStatusText(u"Тестирование завершено",0)
        self.Layout()
    def label_reset(self,event):
        for label in self.label_array:
            label[2].SetLabel("...")
            label[3].SetLabel("...")
            label[4].SetLabel("...")
            label[5].SetLabel("...")
        self.Layout()
    def port_correction(self,event):
        num_ctrl = event.GetEventObject()
        if num_ctrl.GetValue()>num_ctrl.GetMax():
```

```
                num_ctrl.SetValue(num_ctrl.GetMax())
        def demo(self):
            for label in self.label_array:
                if label[0]==u"Скорость 110":
                    label[2].SetLabel("Can’t open")
                    label[3].SetLabel("...")
                    label[4].SetLabel("...")
                    label[5].SetLabel("...")
                elif label[0]==u"Скорость 57600" or label[0]==u"Скорость
115200":
                    label[2].SetLabel("38400")
                    label[3].SetLabel("...")
                    label[4].SetLabel("...")
                    label[5].SetLabel("...")
                else:
                    label[2].SetLabel("OK")
                    label[3].SetLabel("OK")
                    label[4].SetLabel("OK")
                    label[5].SetLabel("OK")
            self.Layout()
    if __name__ == "__main__":
        app = wx.PySimpleApp(0)
        wx.InitAllImageHandlers()
        frame_1 = TestTool(None, -1, "")
        app.SetTopWindow(frame_1)
        frame_1.Show()
        app.MainLoop()
```