

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет магістерської та
аспірантської підготовки
Кафедра інформаційних технологій

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: Розробка інтегрованої інформаційної підсистеми
протипожежного нагляду

ційні

Виконав студент 2 курсу групи МК-61
спеціальності 8.05010101 Інформа-

управляючі системи та технології,
Лозовський Михайло Вікторович

Керівник к.т.н., доцент,
Гнатовська Ганна Арнольдівна

Рецензент к.геогр.н., доцент,
Кузніченко Світлана Дмитрівна

Одеса 2017

АНОТАЦІЯ

Тема магістерської роботи «Розробка інтегрованої інформаційної підсистеми протипожежного нагляду».

Актуальність магістерської роботи полягає в необхідності розробки програмного забезпечення, яке здійснює інтеграцію як різнорідних технічних засобів, так і нових програмних інтерфейсів в єдину комплексну інтегровану систему централізованого нагляду за пожежною безпекою об'єктів.

Об'єкт дослідження – процеси інтеграції та розробки інформаційної підсистеми протипожежного нагляду, що забезпечує швидке реагування пульта охорони на пожежні тривоги, що надходять з об'єктів.

Мета роботи – дослідження та розробка інформаційної підсистеми протипожежного нагляду та реалізація механізму інтеграції підсистеми з комплексною системою централізованого спостереження «Дунай».

Для реалізації поставленої мети були вирішені наступні питання: проведено дослідження та аналіз існуючих систем протипожежної безпеки; проведено огляд сучасних архітектурних рішень пультів централізованого спостереження «Дунай»; визначені вимоги до розробки інформаційної підсистеми протипожежного нагляду; визначена схема взаємодії системи «Дунай» та інтегрованої інформаційної підсистеми протипожежного нагляду; реалізовано механізм інтеграції підсистеми протипожежного нагляду з пультом централізованого нагляду «Дунай», засобами протоколу зв'язку, що визначає формат передачі повідомлень про пожежні тривоги; обрана архітектура та програмні засоби реалізації підсистеми; здійснена програмна реалізація. Практична цінність магістерської роботи полягає в тому, що розроблена інтегрована інформаційна підсистема протипожежного нагляду може бути використана для здійснення спостереження за пожежною безпекою об'єктів.

Ключові слова: інформаційна підсистема, об'єкти охорони, системи протипожежної безпеки, програмне забезпечення.

Магістерська робота містить 68 сторінок, 6 таблиць, 17 рисунків, 13 посилань та 17 листів додатку.

ANNOTATION

The topic of master work Development of the integrated information subsystem of the fire watching".

The relevance of this master work is to develop software that integrates as many technical tools and new software interfaces into a single comprehensive integrated system centralized monitoring of fire safety of objects.

Object of research – the processes of integration and development of information subsystem of the fire supervision, to ensure rapid response of the control of the fire alarm coming from the objects.

Purpose – the research and development of information subsystem of fire monitoring and implementation mechanism of the integration of subsystems with the integrated system of centralized monitoring «Dunay».

To implement this goal have been resolved following objectives: the study and analysis of existing fire safety systems; the review of modern architectural solutions to the remote central monitoring «Dunay»; defined requirements for the development of information subsystem of fire supervision; defines the scheme of interaction of the system «Dunay» and the integrated information subsystem of the fire monitoring; the mechanism of integration of the subsystem fire monitoring Central monitoring of the «Dunay» by means of a communication protocol that defines the format of messages on fire alarm; the selected architecture and software implementation of the subsystem; implemented software implementation.

The practical value of master's work is that developed by the integrated information subsystem of the fire observations, which can be used to monitor fire safety of objects.

Key words: information system, objects of protection, fire security systems, software.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ І УМОВНИХ ПОЗНАЧЕНЬ.....	9
ВСТУП	10
1 ТЕОРЕТИЧНІ ЗАСАДИ РЕАЛІЗАЦІЇ ІНТЕГРОВАНОЇ ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ ПРОТИПОЖЕЖНОГО НАГЛЯДУ	12
1.1 Огляд та аналіз існуючих систем протипожежної безпеки	12
1.2 Призначення та основні функції автоматизованої інформаційної підсистеми протипожежного нагляду	14
1.3 Призначення і можливості комплексної інтегрованої системи централізованої охорони «Дунай»	17
1.4 Архітектура та апаратне забезпечення комплексної інтегрованої системи «Дунай»	20
1.5 Огляд сучасних архітектурних рішень пультів централізованого спостереження КІСЦО «Дунай»	24
1.5.1 Пульт централізованого спостереження малої місткості КІСЦО «Дунай»	25
1.5.2 Пульт централізованого спостереження великої місткості КІСЦО «Дунай» на базі ПЕОМ.....	26
1.6 Програмне забезпечення та архітектура пульта централізованого спостереження	28
2 ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА ФУНКЦІОНУВАННЯ ІНТЕГРОВАНОЇ ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ ПРОТИПОЖЕЖНОГО НАГЛЯДУ	31
2.1 Визначення функціональних можливостей підсистеми протипожежного нагляду.....	31
2.2 Вибір та обґрунтування архітектури підсистеми.....	33
2.3 Вибір інтегрованого середовища розробки інтерфейсу підсистеми	37
2.4 Вибір СУБД	39
2.5 Протокол зв'язку інформаційної системи протипожежного нагляду	42
2.6 Вимоги до інформаційного забезпечення підсистеми протипожежного нагляду.....	43
3 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ ПРОТИПОЖЕЖНОГО НАГЛЯДУ	46
3.1 Проектування бази даних підсистеми протипожежного нагляду	46
3.1.1 Опис таблиць бази даних і зв'язків між ними.....	48

3.2 Проектування функціональних можливостей користувачів підсистеми та інтерфейсу	53
4 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНТЕГРОВАНОЇ ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ ПРОТИПОЖЕЖНОГО НАГЛЯДУ	56
4.1 Реалізація механізму аутентифікації користувачів в системі.....	56
4.2 Реалізація головної форми інформаційної підсистеми	57
4.3 Реалізація механізму інтеграції підсистеми з ПЦС «Дунай»	58
4.4 Обробка пожежних тривог у підсистемі.....	60
4.5 Редагування бази даних	61
4.6 Реалізація механізму пошуку в підсистемі.....	62
4.7 Реалізація журналу обліку подій і тривог.....	63
4.8 Налаштування параметрів інтерфейсу підсистеми.....	65
ВИСНОВКИ.....	68
ПЕРЕЛІК ПОСИЛАНЬ	69
ДОДАТОК А ОСНОВНІ КОДИ ПРОГРАМНИХ МОДУЛІВ.....	71

ПЕРЕЛІК СКОРОЧЕНЬ І УМОВНИХ ПОЗНАЧЕНЬ

Скорочення:

АРМ – автоматизоване робоче місце

АТС – автоматична телефонна станція

КІСЦО – комплексна інтегрована система централізованої охорони

ОПС – охоронно-пожежна сигналізація

ПКП – пультові крайові пристрої

ППК – прибор приймально-контролюючий

ПЦС – пульт централізованого спостереження

СПС – система передачі сповіщень

API – Application Programming Interface – Прикладний програмний інтерфейс

TCP/IP – Transmission Control Protocol (TCP) і Internet Protocol (IP) – набір мережевих протоколів передачі даних, який використовують в мережах

ODBC – Open Database Connectivity – програмний інтерфейс, що забезпечує доступ до бази даних

IDE – Integrated Development Environment – інтегроване середовище розробки програм

RAD –Rapid Application Development – середовище швидкої розробки додатків.

LCL – Lazarus Component Library – бібліотека візуальних компонентів.

ВСТУП

У сучасному суспільстві величезна увага приділяється створенню систем пожежної безпеки об'єктів, які призначені для захисту життя людей і матеріальних цінностей від вогню. Адже небезпека для життя, пов'язана з виникненням пожежі, і збитки, що наносяться вогнем, в десятки разів перевищують ті, які можуть бути викликані крадіжками, пограбуваннями і т.п. Автоматичні системи пожежної сигналізації призначені для швидкого і надійного виявлення пожежі, що зароджується, за допомогою розпізнавання явищ, супроводжуваних пожежою, таких як виділення тепла, диму, невидимих продуктів згорання, інфрачервоного випромінювання і т.п.

Для забезпечення безпеки різних об'єктів (підприємств, торгових приміщень, фірм, державних установ, приватних будинків і квартир) широко застосовуються автоматизовані засоби охорони і пожежного спостереження. Звичайно ці засоби включають встановлені на об'єкті, що охороняється, прилади, одержуючи інформацію від всіх охоронних і пожежних сповіщувачів (датчиків), і пульта централізованого спостереження (ПЦС), що знаходиться в організації, що відповідає за охорону об'єкта, на який сходиться вся інформація про виникаючі тривоги.

Така структура системи найбільш зручна для клієнтів, що користуються послугами автоматизованої охорони, оскільки дозволяє їм не створювати локальні пульти і не організовувати чергування на самому об'єкті. У разі установки автоматизованої системи протипожежної охорони на великому об'єкті, наприклад, будівлі, коли вона охоплює велику кількість приміщень, локальний пульт нагляду може застосовуватися спільно з підключенням до пульта централізованого спостереження охоронно-протипожежної організації, але для маленьких об'єктів, наприклад, окремого офісу, що складається з декількох невеликих приміщень, це недоцільно.

Надходження повідомлень про тривоги безпосередньо особі, що несе чергування на пультах централізованого спостереження охоронно-пожежної організації, гарантує його негайну обробку і вживання негайних заходів по запобіганню та виявленню пожежі.

Оскільки сьогодні існує великий спектр технічних засобів протипожежної сигналізації, від старої апаратури, що добре зарекомендувала себе, вже встановленої на великій кількості об'єктів, до новітніх приладів, пропонованих вітчизняними і зарубіжними фірмами, важливе завдання пульта централізованого спостереження – інтеграція різномірних технічних засобів в єдину

комплексну інтегровану систему централізованої охорони (КІСЦО). Така КІСЦО окрім можливості фізичного підключення існуючої техніки до себе повинна забезпечити ще і можливість вбудовування в своє програмне забезпечення програмних інтерфейсів нових приладів. При цьому вся апаратура, що вже реально функціонує в рамках даної конкретної системи, продовжує працювати так само як і раніше. «Дунай» – перша така система, розроблена в Україні. Вона задумана і створена колективом науково-виробничої фірми «Венбест» [1].

Системи охоронної сигналізації, що підключені до пульта централізованого спостереження, давно застосовуються державними службами (міліція, ДСО), а сьогодні і приватними організаціями, зайнятими у сфері охорони, а державна пожежна охорона не має таких систем і їх створення, що включає установку на об'єктах приладів приймально-контрольних (ППК), розгортання ПЦС і виділення каналів зв'язку ППК з пультом централізованого спостереження зажадає великих витрат. Але існує можливість без зайвих зусиль здійснити підключення пожежної сигналізації до апаратури, що вже задіяна в охороні об'єкту. При цьому повідомлення про пожежні тривоги поступають на пульт централізованого спостереження охоронної організації, і повинні бути передані черговому пожежної охорони по телефону.

Значно зручніше було б, щоб тривоги поступали та були оброблені безпосередньо на комп'ютерний пульт в пожежній охороні. Програмне забезпечення сучасних пультів охоронної сигналізації мають функцію, яка автоматично проводить фільтрування та відокремлення повідомлень про пожежні тривоги із загального потоку повідомлень від устаткування на об'єктах і здійснює передачу їх по каналам зв'язку на пульт пожежної охорони.

Метою магістерської роботи є проектування та програмна реалізація інтегрованої підсистеми протипожежного нагляду, що реалізовує функції видаленого робочого місця чергового пожежної охорони. Ця підсистема є підсистемою КІСЦО «Дунай», яка встановлюється на робочому місці чергового пожежної охорони і робить пульт централізованого спостереження «Дунай» розподіленим.

1 ТЕОРЕТИЧНІ ЗАСАДИ РЕАЛІЗАЦІЇ ІНТЕГРОВАНОЇ ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ ПРОТИПОЖЕЖНОГО НАГЛЯДУ

1.1 Огляд та аналіз існуючих систем протипожежної безпеки

Пожежна сигналізація є одним з найважливіших елементів системи попередження пожежі, основне завдання якої – оперативне виявлення спалаху і негайна передача відповідного сигналу на пульт централізованого спостереження. Автоматична система пожежної сигналізації – це технічно складна система, що включала в свій склад:

- апаратуру, яка дозволяє виявити джерело виникнення пожежі (пожежна сигналізація);
- пристрої автоматичного включення: мовного оповіщення, системи пожежогасіння, димовидалення;
- подачу керуючих сигналів на систему контролю і управління доступом.

Система автоматичної пожежної сигналізації, це система, що одержує, обробляє, передає і формулює в заданому вигляді інформацію про загорання на території, що охороняється. У цьому короткому визначенні розкриті всі основні функції автоматичної пожежної сигналізації.

Приміщення, обладнані системою автоматичної пожежної сигналізації забезпечують збереження майна організації, а також життя і здоров'я людей у разі виникнення пожежі. Будь-яка позаштатна ситуація, будь то електричне замикання чи пожежа через необережне поводження з вогнем, може бути вирішена без безпосередньої участі клієнта: пожежні, які отримали тривожний сигнал, прибудуть на місце, не чекаючи дзвінка.

Швидке і надійне функціонування автоматичної пожежної сигналізації забезпечується за допомогою сучасних надчутливих датчиків, які гарантують миттєве оповіщення відповідних інстанцій.

Сучасні технології дозволяють системі залишатися несприйнятливою до впливу зовнішніх факторів, що відрізняються від факторів пожежі, що стало можливим завдяки індикаторам температури, світла і тепловим датчикам. Якщо ж прилади зафіксували спалах, система автоматично приступає до пожежогасіння та димовидалення [2].

Незалежно від типу пожежної сигналізації, в її склад, як правило, входять наступні елементи (рис. 1.1):

- приймально-контрольний прилад – пристрій, який обробляє і аналізує інформацію, що надходить від пожежних датчиків;
- пожежні сповіщувачі – чутливі прилади, що фіксують: дим, підвищення температури, відкрите полум'я. Встановлені по всьому приміщенню.
- система оповіщення про пожежу – технічний засіб сигналізації, призначене для оповіщення людей про пожежу; види сповіщувачів: світлові, звукові, мовні, комбіновані (світлозвукові);
- пристрій зв'язку – передає сигнали на пульт централізованого спостереження за допомогою різних каналів зв'язку – телефонної лінії, радіоканалу, GSM і інших.
- світлове табло «Вихід» – призначене для встановлення всередині приміщень з метою світлової вказівки евакуаційних місць виходу під час пожежі та інших надзвичайних ситуаціях;
- резервне джерело живлення – призначене для забезпечення безперебійним живленням систем охоронно-пожежної сигналізації.



Рисунок 1.1 – Состав устаткування пожежної сигналізації

При виявленні пожежними датчиками джерела виникнення пожежі (задимлення, відкрите полум'я або різке збільшення температури) в приміщенні, що охороняється, включається виконання закладеного в систему автоматичної пожежної сигналізації алгоритму дій. При визначенні загоряння сигнал з пристрою зв'язку надходить на пульт міністерства надзвичайних ситуацій (МНС).

1.2 Призначення та основні функції автоматизованої інформаційної підсистеми протипожежного нагляду

Охоронно-пожежна сигналізація (ОПС) – це одержання, обробка, передача і уявлення в заданому вигляді споживачам інформації про проникнення на об'єкти, що охороняються і пожежі на них за допомогою технічних засобів. Споживачем інформації є персонал, на який покладено функції реагування на тривожні і службові повідомлення, що надходять з об'єктів, що охороняються.

Об'єктом, що охороняється називається окреме приміщення, яке містить матеріальні чи інші цінності, обладнане технічними засобами ОПС, або комплекс приміщень, розосереджених в межах одного або декількох будівель, об'єднаних загальною територією і охоронюваних підрозділами охорони. Місця можливого проникнення на ГО або окремі зони, що охороняються обладнуються різними сповіщувачами, які включаються в шлейф сигналізації.

Повідомленням в техніці ОПС називається повідомлення, що несе інформацію про контрольовані зміни стану об'єкту, що охороняється або технічного засобу ОПВ і передане за допомогою електромагнітних, електричних, світлових і (або) звукових сигналів. Повідомлення діляться на тривожні і службові. Тривожне повідомлення містить інформацію про проникнення пожежі, службове – про «взяття» під охорону, «зняття» з охорони, несправності апаратури та ін.

Комплекс охоронно-пожежної сигналізації – це сукупність спільно діючих технічних засобів охоронної, пожежної і (або) охоронно-пожежної сигналізації, встановлених на об'єкті, що охороняється і об'єднаних системою інженерних мереж і комунікацій [1].

Система передачі сповіщень (СП) – це сукупність спільно діючих технічних засобів для передачі по каналах зв'язку і прийому в пункті централізованої охорони сповіщень про проникнення на об'єкти, що охороняються по-

жежі, засобами службових та контрольно-діагностичних повідомлень, а також для передачі та прийому команд управління.

Пункт централізованої охорони (ПЦО) – це диспетчерський пункт централізованої охорони ряду розосереджених об'єктів від проникнення і пожежі з використанням СПП.

Технічні засоби (ТЗ) охоронної та охоронно-пожежної сигналізації, призначені для отримання інформації про стан контрольованих параметрів на об'єкті, що охороняється, приймання, перетворення, передачі, зберігання, відображення цієї інформації у вигляді звукової та світлової сигналізації та класифікується за двома ознаками: області застосування і функціональному призначенню.

По області застосування ТЗ діляться на охоронні, пожежні та охоронно-пожежні; за функціональним призначенням – на технічні засоби виявлення (сповіщувачі), призначені для отримання інформації про стан контрольованих параметрів і ТЗ оповіщення, призначені для прийому, перетворення, передачі, зберігання, обробки і відображення інформації.

Призначення системи пожежної сигналізації визначає її загальну структуру, а саме, наявність трьох складових системи, що виконують різні функції [1]:

- виявлення пожежі здійснюється автоматичними пожежниками сповіщувачами з різними принципами виявлення і різними методами обробки і обміну інформацією;
- обробка інформації, що поступає з сповіщувачів, і видача результатів операторові виконуються центральною станцією і пультом управління;
- виконання, певних дій для сповіщення персоналу і пожежної частини для усунення вогнища пожежі, виконується центральною станцією;
- швидке і точне реагування підрозділів пожежної частини і локальних постів пожежної охорони.

Всі три складових тісно взаємозв'язані між собою, і ефективність роботи системи пожежної сигналізації в цілому залежить від надійності і стабільності роботи кожної її складової.

Проте, основну роль при створенні професійних систем пожежної безпеки об'єктів відіграють пожежні сповіщувачі. Саме вони повинні забезпечити швидке і надійне виявлення вогнища пожежі.

Автоматизована система зв'язку й оперативного управління протипожежного нагляду призначена для удосконалювання діяльності підрозділів пожежної охорони шляхом автоматизації вирішення завдань управління, прийняття рішень і оптимізації існуючої організаційної структури системи управління і входних у неї підрозділів.

Автоматизована інформаційна підсистема протипожежного нагляду призначена для:

- прийому заявок про пожежі, що надходять по каналах зв'язку або безпосередньо від заявників;
- прийому електричних сигналів від систем пожежного спостереження;
- мобілізації спеціалізованих служб на ліквідацію пожежі;
- забезпечення необхідною інформацією пожежно-рятувальних підрозділів;
- забезпечення необхідною інформацією підрозділів, що працюють на місці пожежі;
- забезпечення оперативного зв'язку між персоналом управління ліквідацією пожежі;
- збору, хронометрування, сортування, накопичення, збереження, відображення, документування і видачі зацікавленим відомствам відомостей про пожежі.

Використання сучасних каналів зв'язку і передачі даних спільно з обладнанням дозволяє відмовитися від стандартних принципів надання охоронних послуг, розширюючи можливості реалізації. Схеми організації охоронних комплексів можуть ґрунтуватися на внутрішньо об'єктовій системі передачі сповіщень.

Сотова охоронно-пожежна сигналізація, яка використовує GSM зв'язок призначена для оповіщення пультів охорони, власників будинків, дач, квартир, торговельних точок і ін. про пожежу або несанкціоноване проникнення.

За допомогою коротких SMS-повідомлень система сигналізує про спрацьовування на об'єкті охоронних і пожежних датчиків. В разі потреби SMS-повідомлення можуть бути продубльовані голосовим повідомленням [1].

Схема організації охоронного комплексу з використанням GSM зв'язку з пультом охорони (рис. 1.2).

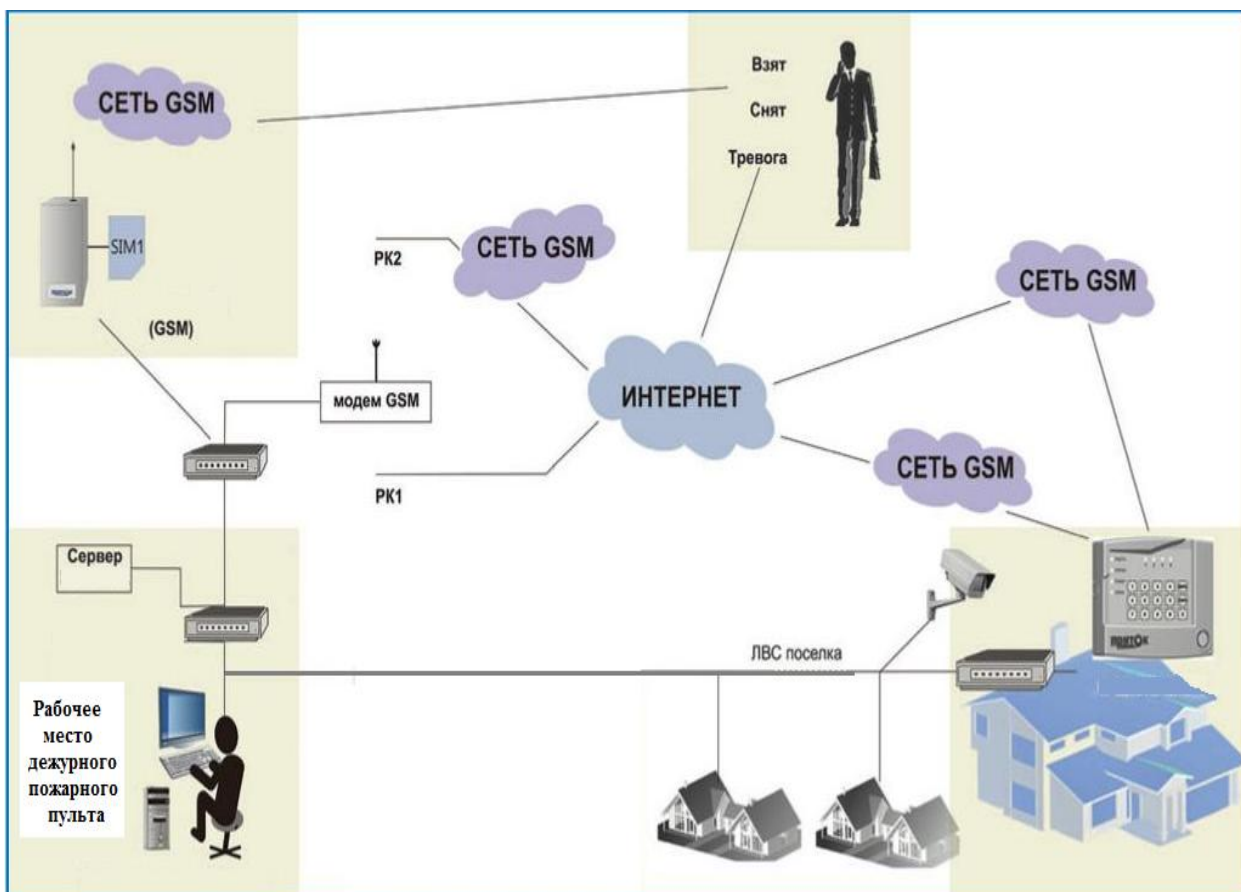


Рисунок 1.2 – Схема організації охоронного комплексу з використанням GSM-зв'язку з пультом охорони

1.3 Призначення і можливості комплексної інтегрованої системи централізованої охорони «Дунай»

Сучасна практика показує, що найбільш перспективним і раціональним шляхом організації охорони є застосування інтегрованих систем, які включають в себе кілька напрямків технічних засобів, що взаємодіють між собою в єдиній базі даних або в складі єдиного комплексу, що складається з декількох підсистем. Однією з таких інтегрованих систем, успішно застосовується в підрозділах охорони з 1991 року, стала інтегрована система централізованої охорони «Дунай» [2].

Пожежний моніторинг – це передача сигналів від пожежних датчиків на пульт відділів міністерства надзвичайних ситуацій. КІСЦО «Дунай» призначена для цілодобового спостереження за станом об'єктів, квартир, банків, готелів, офісів і т.п. від несанкціонованого проникнення людини або виникнення пожежі, для екстреного виклику невідкладної допомоги або служби

охорони, а також для прийому і передачі сигналів керування і контролю стаціонарних і пересувних об'єктів [1]. Система «Дунай» забезпечує:

- збір і обробку сигналів, що поступають від об'єктів, формування і видачу на дисплей пульта централізованого спостереження тривожної, службової, довідкової і графічної інформації;
- передачу інформації по лініях зв'язку і каналах радіозв'язку, зокрема по каналах «Пакетної радіомережі»;
- автоматичне узяття (зняття) об'єктів під спостереження за допомогою електронного замку, клавіатури або електронної картки TOUCH MEMORY з видачею користувачу відповідного підтвердження;
- можливість роботи з об'єктовим устаткуванням з «ручною» тактикою охорони;
- можливість дублювати інформацію по інших каналах зв'язку;
- високі діагностичні властивості блоків і вузлів системи;
- повну інформативність, достовірність і надійність інформації, що передається;
- резервування всіх вузлів і блоків системи від знеструмлення живлячої мережі 220 В;
- повне збереження оперативної інформації в незалежній пам'яті приладів при будь-яких нештатних ситуаціях в системі.

Система «Дунай» складається з:

- пульта централізованого спостереження;
- пультавих крайових пристроїв;
- програмного забезпечення;
- устаткування, що ретранслює інформацію від об'єктових приладів на пульт централізованого спостереження;
- об'єктових приладів з різними протоколами функціонування;
- сервісного устаткування.

Принципи побудови КІСЦО «Дунай» [1] дозволяють користувачу дістати дуже великі можливості по конфігурації комплексу технічних засобів безпеки.

Архітектура комплексної інтегрованої системи централізованого спостереження «Дунай», як системи збору і обробки інформації, в переважній більшості є класичною триланковою структурою:

- ПЦС (пульт централізованого спостереження);

- АТС (автоматична телефонна станція);
- об'єкт.

Пульт централізованого спостереження – це ланка структури, призначена для збору і обробки інформації тієї, що поступає від інших ланок. У системі «Дунай» пульт централізованого спостереження, звичайно, є ПЕОМ, до якої, через інтерфейси RS-232 підключаються пульти крайові пристрої (ПКП) різних типів, які призначені для роботи з устаткуванням іншої ланки структури (рис. 1.3).

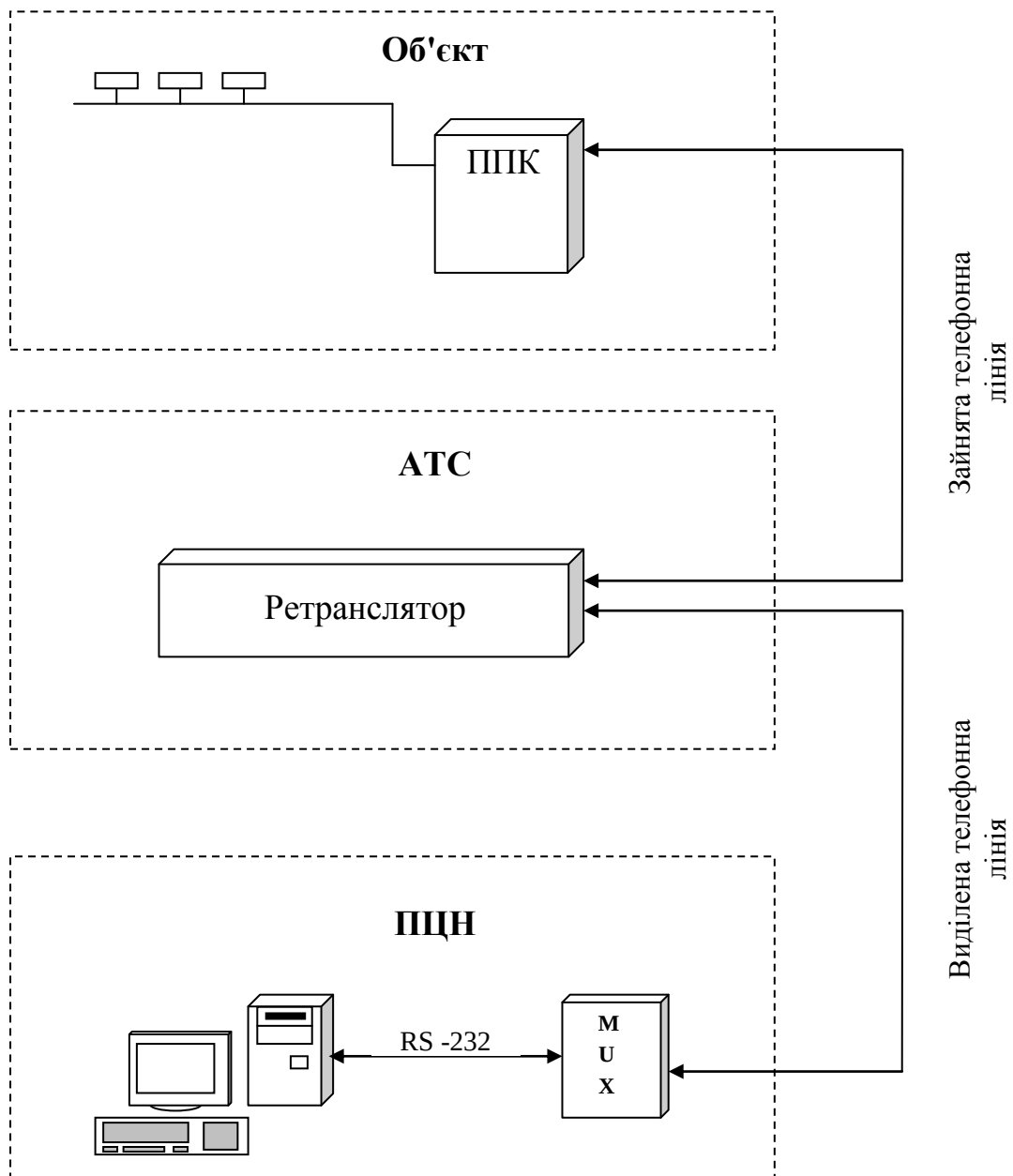


Рисунок 1.3 – Стандартна схема застосування ППС різного типу до КІСЦО «Дунай»

Об'єкт – це ланка структури КІСЦО «Дунай» призначена для безпосереднього збору інформації з датчиків про їх стан і для здійснення передачі цієї інформації іншим ланкам.

АТС – це ланка структури, призначена для передачі інформації від різноманітного устаткування ланки «Об'єкт» до устаткування ланки ПЦС. В деяких випадках дана ланка може бути відсутньою. При територіальному розподілі устаткування «Об'єктів» устаткування «АТС» виконує роль локального об'єднання, збору і трансляції інформації на ПЦС.

1.4 Архітектура та апаратне забезпечення комплексної інтегрованої системи «Дунай»

Відкрита архітектура комплексної інтегрованої системи централізованої охорони «Дунай» дозволяє використовувати різне устаткування і схеми його підключення, але найчастіше застосовується стандартна схема з трьох ланок (рис. 1.4).

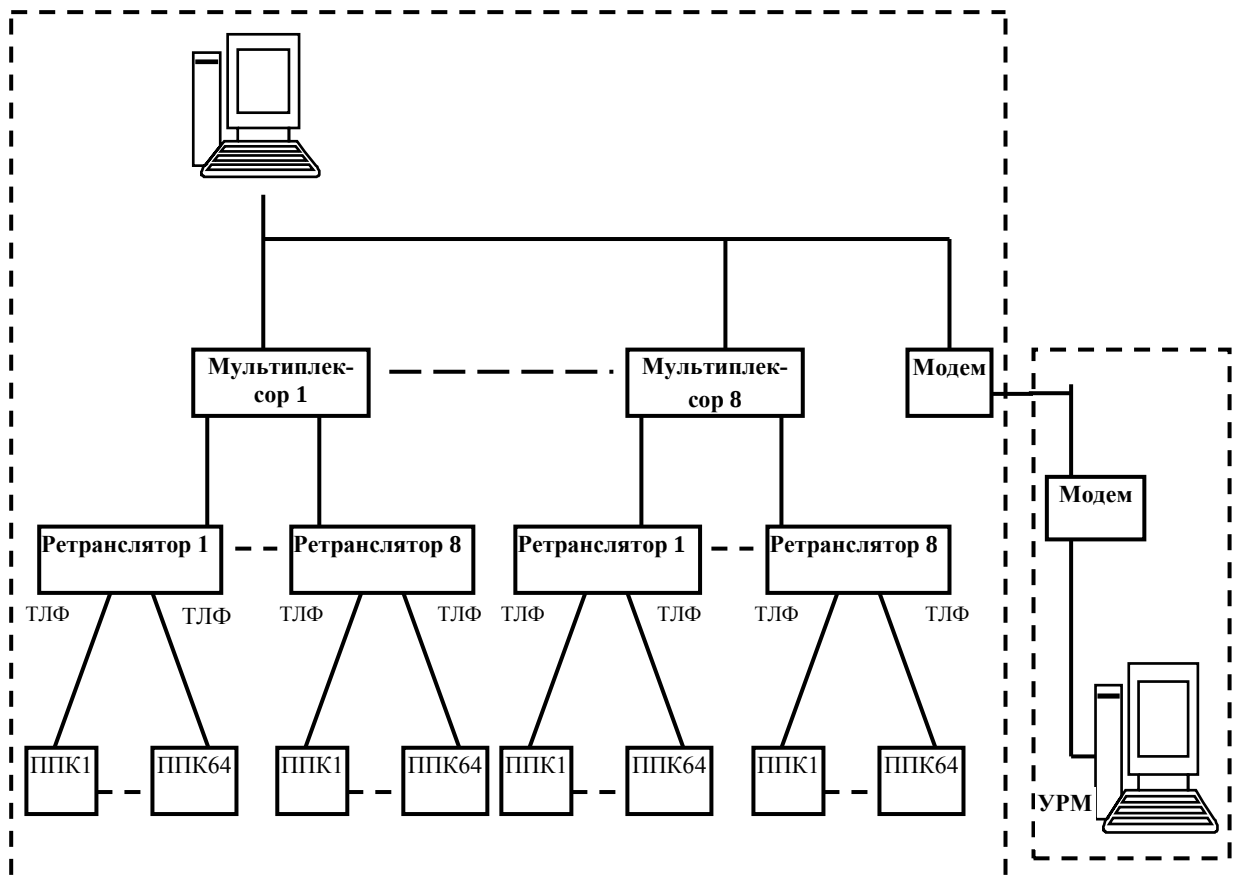


Рисунок 1.4 – Структурна схема взаємодії КІСЦО «Дунай» та інтегрованої інформаційної підсистеми протипожежного нагляду

У цій схемі як пультові крайові пристрої, сполучені з ПЕОМ, використовується мультиплексор SPIN MUX, до якого по виділених лініях зв'язку можна підключити до восьми комплектів ретрансляторів SPIN ATU, що встановлюються на міських автоматичних телефонних станціях. До кожного комплекту ретрансляторів можливо підключити по зайнятій телефонній лінії до 512 різноманітних приладів встановлюваних на об'єктах [1, 2].

Крім стандартної схеми підключення устаткування до комплексної інтегрованої системи централізованої охорони, можливі інші способи організації зв'язку з пультом централізованого спостереження залежно від вживаних пристроїв і їх функціональних можливостей:

- ретранслятор SPIN ATU може використовуватися як пультівий крайовий пристрій без застосування при цьому мультиплексора SPIN MUX. В цьому випадку ланка АТС виключається, а об'єктове устаткування, як і в стандартній схемі, по зайнятих або виділених лініях зв'язку підключається до ретранслятора;
- якщо на об'єкті встановлені ППК SPIN URB, здатні працювати по радіо каналу, інформація може передаватися з них на такий же прилад, встановлений на пульті централізованого спостереження;
- система «Дунай» може працювати з пакетною радіомережею. При цьому як апаратура, що утворює канал зв'язку застосовується модем РАДИО-СРП. Як об'єктове устаткування можуть використовуватися різні ППК, що забезпечують взаємодію з модемами РАДИО-СРП в протоколі X.25. Така схема зазвичай застосовується на місцевості де вже розгорнена пакетна радіомережа.

Мультиплексор каналів SPIN MUX призначений для сполучення ПЕВМ пульта централізованого спостереження по стику RS-232 з одного боку і каналів зв'язку, що сполучають до восьми видалених ретрансляторів з іншого боку. Зв'язок між мультиплексором і ПЕОМ здійснюється із швидкістю 9600 бит/с. Мультиплексор може бути підключений кабелями до двох комп'ютерів, і перемикання для роботи з вибраним комп'ютером здійснюється у ручному режимі. Спосіб передачі повідомлень по телефонному каналу – амплітудна модуляція, швидкість передачі сповіщень 200 бод. Прилад SPIN MUX приймає від ретрансляторів пакети про стан об'єктів, що охороняються, і встановлені на них технічні засоби охорони і передає їх в ПЕВМ пульта централізованого спостереження. Також він передає команди, що здійснюють управління з ПЦС на ретранслятор. Цей прилад здійснює контроль за справністю каналів зв'язку на ділянці «ретранслятор – ПЦС». У разі порушення

зв'язку з ретранслятором мультиплексор SPIN MUX передає на ПЕОМ пульта централізованого спостереження спеціальне сповіщення про це, а у разі порушення зв'язку з ПЕВМ – включає вбудований звуковий сигнал, що оповіщає черговий персонал про несправність [1].

Мультиплексор складається з наступних функціональних частин:

- процесорного вузла, в який входять:
- однокристальний мікроконтролер;
- пристрій ППЗУ;
- незалежне ОЗУ;
- прийомо-передавач сигналів послідовного інтерфейсу RS-232;
- прийомо-передавач сигналів з ретрансляторами;
- вузла електроживлення;

Як проміжна ланка в структурі передачі сповіщень з об'єкту на пульт централізованого спостереження, при використанні телефонних ліній, застосовуються Ретранслятори SPIN ATU. Вони встановлюються на АТС або на об'єктах при організації локальних ПЦС. Ретранслятор призначений для сполучення каналів зв'язку, що сполучають видалені ППК з одного боку, з мультиплексором SPIN MUX з іншого боку; реалізації протоколу автоматизованої і ручний тактик охорони на ділянці «ретранслятор – об'єкт»; прийому–передачі сповіщень від ППК, встановлених на об'єктах; прийому–передачі пакетів від (на) ПЦН (мультиплексор SPIN MUX); контролю за справністю каналів зв'язку на ділянці «ретранслятор – об'єкт» і видачі на пульт централізованого спостереження сповіщень про несправність; забезпечення оптимізації часу циклу опиту залежно від кількості приписаних з ПЦС приладів приймально-контрольних.

Функціонально ретранслятор є універсальним приладом, обслуговуючим до 512 телефонних ліній (напрямів). Конструктивно до одного корпусу ретранслятора може бути підключено до 64 телефонних ліній (напрямів). Місткість ретранслятора збільшується до максимальної шляхом установки додаткових корпусів ретрансляторів (без модулів управління) і з'єднання їх з першим ретранслятором стрічковим кабелем, що входить в комплект. Знов встановлені ретранслятори працюють під управлінням першого ретранслятора [1].

Роботою ретранслятора управляє однокристальний мікроконтролер за програмою, що зберігається в ППЗУ. Мікроконтролер, ППЗУ і інші елементи управління роботою ретранслятора розміщені на модулі SPIN ATU/CPU. Для сполучення з лініями зв'язку, залежно від тактики охорони, в корпус ретранс-

лятора може бути встановлено до 8-ми модулів сполучення. Один модуль підтримує роботу 8-ми напрямів.

Модуль SPIN ATU/MF реалізує принцип роботи частотного ущільнення лінії зв'язку при роботі з ППК типу SPIN VB, комунікатором SPIN KM, ППК системи «Каштан» і груповими концентраторами АСЦН «Комета».

Модуль SPIN ATU/LIN реалізує принцип роботи струмової петлі в лінії зв'язку з ППК типу «Дніпро», «Сигнал-41» і аналогічними їм приладами.

Модуль SPIN ATU/ATL реалізує принцип роботи частотного ущільнення лінії зв'язку з частотою 18 кГц (прилади типу «Атлас»).

Окрім підключення до мультиплексора SPIN MUX по виділеній лінії зв'язку, ретранслятор SPIN ATU може бути підключений:

- до мультиплексора системи «Каштан» по виділеній лінії зв'язку;
- до ПЕОМ ПЦС через модеми «Радіо СРП»;
- до ПЕОМ ПЦС через модеми типу ZYXEL, MOTOROLA, MULTITECH і інші по виділених або зайнятих лініях зв'язку;
- до радіо-мультиплексора SPIN PCO (використовуючи радіо-НС) або до ПЕОМ ПЦС через послідовний порт RS-232 на відстань до 15 м. При використанні «конвертерів» сигналів стику RS-232 – до 1 км.;
- одночасно до мультиплексора SPIN MUX або мультиплексору системи «Каштан» по виділеній лінії зв'язку і через послідовний порт RS-232 до ПЕОМ ПЦС або контрольної панелі SPIN CP.

Одним з найважливіших елементів системи «Дунай» є об'єктове устаткування – ППК або централь. Це пояснюється декількома причинами. По-перше, ППК – це прилади, з якими мають справу безпосередньо користувачі, тому вони повинні, володіючи всією необхідною функціональністю, бути достатньо простими для використання і настройки користувачем; по-друге, саме від цього приладу залежить ухвалення рішення або про реальне проникнення на об'єкт, або про просте відхилення параметрів шлейфів сигналізації від норми; по-третє, саме об'єктовий прилад на 99% визначає вартість всієї системи в цілому, оскільки це наймасовіший прилад в системі централізованого спостереження [3].

У системі «Дунай» можуть бути використані різні види ППК, сигналізації, що відрізняються по максимальній кількості шлейфів, що підключаються, часу роботи від резервного джерела живлення за відсутності або неправильних параметрах напруги побутової електромережі, можливості роботи в мережі з іншими ППК для забезпечення охорони великої кількості приміщень, ціні.

Якщо не вимагається підключення до системи «Дунай» вже наявного об'єктового устаткування, на об'єктах звичайно встановлюються сучасні прилади «ВБД-4», «ВБД-6», «Дунай», «SPIN VB» [1, 3].

Будь-який прилад приймально-контрольний (ППК) виконує наступні функції:

- збір інформації від датчиків, що знаходяться на шлейфах, підключених до ППК і формування сповіщення про тривогу при відхиленні сигналу від стандартних параметрів;
- передача сповіщень про тривоги на пульт централізованого спостереження через устаткування зв'язку (ретранслятор, мультиплексор каналів) або на локальний пульт спостереження; електроживлення від мережі змінного струму напругою 220 В і автоматичне перемикання на роботу від встановленого в корпус ППК резервного джерела (акумулятора) у разі неможливості роботи від мережі.

Конфігурація і режими роботи ППК програмуються за допомогою спеціальної клавіатури або персонального комп'ютера.

Додатковими можливостями для приладів приймально-контрольних є: можливість підключення інших ППК для організації мережі на об'єкті, що охороняється, подача сигналу на світлові і звукові оповісники у разі виникнення тривоги, виведення повідомлень про зафіксовані ППК події на принтер, що підтримує протокол Centronics.

1.5 Огляд сучасних архітектурних рішень пультів централізованого спостереження КІСЦО «Дунай»

Застосування різних підсистем комплексної інтегрованої системи «Дунай» окремо або в складі комплексу дозволяє вирішувати завдання не тільки охорони і пожежної безпеки, а й здійснювати спостереження за станом самих об'єктів охорони, запобігаючи аваріям і техногенним подіям.

Структура інтегрованої системи «Дунай» така, що один пульт охорони, створений на її основі, забезпечує охорону (моніторинг) невеликого або великого об'єкта і безлічі об'єктів одночасно. ІС «Дунай» може впроваджуватися послідовно, починаючи з однієї необхідної підсистеми [2].

Моніторинг проводиться з будь-якої кількості автоматизованих робочих місць (АРМ), встановлених в приміщеннях служби безпеки, технічного персоналу, що супроводжує і обслуговує систему, а також на «віддалених АРМ», встановлених на будь-якій відстані від пульта охорони.

Використання сучасних каналів зв'язку і передачі даних спільно з обладнанням «Дунай» дозволяє відмовитися від стандартних принципів надання охоронних послуг, розширюючи можливості реалізації. Схеми організації охоронних комплексів можуть ґрунтуватися на об'єктовій системі передачі сповіщень.

Залежно від кількості (місткості) контрольованих об'єктових приладів змінюється побудова пультів централізованого спостереження і застосування на ньому того або іншого устаткування.

З міркувань економічної доцільності пульти малої місткості (до 200 – 300) об'єктів будуються на базі контрольної панелі SPIN CP [1]. Такі пульти можуть працювати на вибір, або по радіоканалу, або по телефонних лініях.

При збільшенні місткості пульта він будується на базі персональної ЕОМ. При цьому кількість автоматизованих робочих місць ПЦС може змінюватися від одного шістнадцяти.

Програмне забезпечення комплексу дозволяє гнучко розподілити повноваження автоматизованих робочих місць з метою виконання різних функцій операторами системи «Дунай».

1.5.1 Пулт централізованого спостереження малої місткості КІСЦО «Дунай»

Контрольна панель виконує роль ПЕОМ на пульті централізованого спостереження малої місткості. Її застосування знижує витрати на устаткування малих ПЦС, що реалізують автоматизовану тактику охорони [2].

Контрольна панель є виконаною з застосуванням SMD-технології – модуль на базі однокристального мікропроцесора. Для управління роботою панелі використовується стандартна клавіатура для комп'ютерів. Контрольна панель виконує наступні функції:

- прийому, обробки і зберігання інформації з об'єктів;
- формування даних для відображення на рідкокристалічному дисплеї з підсвічуванням (4 рядки по 20 знаків);
- формування даних для виводу на принтер;
- звукового сповіщення оператора про прийняті панеллю сповіщення;
- звукового сповіщення про тривоги, що зберігаються в буфері панелі, не оброблені оператором.

Пульт централізованого спостереження малої місткості на базі контрольної панелі SPIN CP дозволяє підключити один з наступних пультових крайових пристроїв:

- мультиплексор каналів SPIN MUX, що дозволяє сполучати з контрольною панеллю ретранслятори SPIN ATU;
- ретранслятор SPIN ATU по стику RS-232;
- радіо мультиплексор SPIN PCO, що дозволяє організовувати роботу радіомережі, за допомогою якої можна охороняти до 512 об'єктів. Прийом – передача сповіщень може бути організована через ретранслятори на базі приладу SPIN URB, збільшуючи таким чином радіус дії підсистеми для охорони об'єктів;
- радіо мультиплексор SPIN PCO, що дозволяє організовувати роботу радіомережі, в яку може бути включено до 8 ретрансляторів SPIN ATU.

Контрольна панель може бути застосована для організації локального ПЦС, що реалізовує автоматизовану тактику охорони. В цьому випадку контрольна панель підключається по стику RS-232 до ретранслятора SPIN ATU. При цьому ретранслятор може бути одночасно сполучений по виділеному каналу з мультиплексором каналів SPIN MUX іншого ПЦС, а інформація, що буде передана ретранслятором, при необхідності, може дублюватися на обидва пульта централізованого спостереження.

1.5.2 Пульт централізованого спостереження великої місткості КІСЦО «Дунай» на базі ПЕОМ

Сучасний розвиток і розповсюдження комп'ютерної техніки зумовили використання ПЕОМ для реалізації пультів централізованого спостереження великої місткості, оскільки вона володіє великим об'ємом постійної (дискової) і оперативної пам'яті для зберігання всієї необхідної інформації, зручним графічним пристроєм виведення інформації (дисплей), при цьому всі функції, передбачені вимогами до пульта централізованого спостереження великої місткості реалізуються програмним забезпеченням, що дозволяє досягти найбільшої гнучкості системи [1].

Пульт централізованого спостереження великої місткості «Дунай» в загальному випадку може складатися з:

- двох ПЕОМ типу IBM PC (у разі використання мережевого варіанту – до 16 комп'ютерів)
- принтера (єдиного для мережі);
- джерела безперебійного живлення типу SMART UPS;
- до 8-ми мультиплексорів каналів SPIN MUX, що дозволяють підключити до однієї ПЕОМ від 1 до 64 ретрансляторів SPIN ATU;
- до 8-ми мультиплексорів каналів SPIN MUX, що дозволяють підключити до однієї ПЕОМ від 1 до 64 ретрансляторів SPIN ATU;
- до 8-ми радіо мультиплексорів SPIN PCO, що дозволяють організувати роботу радіомережі, зокрема через маршрутизатори, за допомогою якої може охоронятися до 512 об'єктів в одній радіомережі;
- до 8-ми радіо мультиплексорів SPIN PCO, що дозволяють організувати роботу 8-ми радіомереж, в кожній з яких може бути включено до 8-ми ретрансляторів SPIN ATU;
- до 8-ми модемів «Радіо СРП», що забезпечують прийом – передачу сповіщень в середовищі пакетної радіомережі від централей SPIN URB через модем «Радіо СРП» або від ретрансляторів SPIN ATU;
- до 8-ми базових комплектів SUR-GARD, забезпечуючих прийом сповіщень від видалених централей;
- поєднання перерахованих пультавих крайових пристроїв;
- програмного забезпечення.

Використання що найменш двох комп'ютерів передбачене тому, що один з них працюватиме в режимі «майстра» (тобто виконує завдання прийняття сповіщення від апаратури), інший – в режимі «клієнта» (у «гарячому резерві»), повністю дублюючи інформацію комп'ютера-майстра.

Працювати з апаратурою (обробляти сповіщення, знімати з охорони, брати під охорону, проводити опит, редагувати базу даних і ін.) можна на будь-якому комп'ютері. Таким чином, виходить два робочих місця, але на комп'ютері-клієнті, без припинення охорони, можна вийти з програми «Дунай» і запустити іншу програму з пакету «Дунай», працювати в інших програмах або вимкнути його, якщо він не потрібен (але на якийсь час, коли немає адміністратора, другу машину краще тримати включеною з програмою «Дунай», тобто в «гарячому резерві»). При виході з ладу комп'ютера-майстра, комп'ютер-клієнт автоматично переходить в режим майстра і про-

довжує охорону, зберігаючи всю інформацію, яка була на комп'ютері-майстра.

Кожен мультиплексор слід підключити на два комп'ютери і встановити перемикач, що виконує комутацію всіх мультиплексорів з одного комп'ютера на інший. Таким чином, у разі виходу з ладу комп'ютера-майстра, користувачу необхідно лише перемкнути одним перемикачем всі мультиплексори на резервний комп'ютер, який, тим часом, автоматично переходить в режим майстра [2, 3].

Система «Дунай» підтримує роботу з мультиплексорами в кількості до 8 одиниць. Кількість робочих місць (комп'ютерів) передбачена від 2 до 16 одиниць. З цих комп'ютерів необхідно вибрати два комп'ютери, які матимуть можливість працювати з апаратурою, тобто мультиплексорами. Вони повинні мати кількість послідовних портів (СОМ-портів), що дозволяє підключати потрібну кількість мультиплексорів (по одному мультиплексору на порт), при необхідності, можна встановити плату розширення СОМ-портів.

Кожен мультиплексор повинен бути підключений до двох комп'ютерів (майстра і резервного клієнта), враховуючи, що кожен мультиплексор необхідно підключати на однаковий номер СОМ-порту кожного з двох комп'ютерів (щоб уникнути плутанини). Решта комп'ютерів працюватиме в режимі клієнтів, «спілкуючись» з апаратурою (мультиплексорами) засобами локальної мережі.

Таким чином, вважаємо, що ПЕОМ-1 і ПЕОМ-2 апаратно взаємозамінні, тобто і ПЕОМ-1 і ПЕОМ-2 може працювати в режимі майстра (але у нинішній момент одна з цих машин працює майстром, інша, в режимі клієнта, в гарячому резерві повністю дублюючи інформацію першої).

1.6 Програмне забезпечення та архітектура пульта централізованого спостереження

Програмне забезпечення (ПО) пульта комплексної інтегрованої системи «Дунай» призначено для інформаційного і функціонального забезпечення дій користувачів системи. Структура ПО і дружній інтерфейс користувача орієнтовані на те, щоб в системі мали змогу функціонувати різні прилади. Програмне забезпечення системи підтримує роботу приладів по телефонних лініях, радіоканалу, автодозвону, автоматично або з застосуванням «ручної» тактики пересилає sms-повідомлення на мобільні пристрої відповідальним особам (госпорганам) і групам затримання [2].

З урахуванням незамкнутої архітектури апаратних засобів і досить універсального підходу в програмному забезпеченні вирішене завдання отримання відкритої системи за допомогою створення драйверів апаратури з відкритим інтерфейсом. Всі рішення, які реалізовані в програмному забезпеченні пульта централізованого спостереження, направлені на досягнення головної мети – дати можливість оператору (черговому) максимально швидко і безпомилково реагувати на сповіщення, що поступили. Багато в чому це досягається наочністю і продуманістю до дрібниць призначеного для користувача інтерфейсу.

Програмне забезпечення, яке функціонує у КІСЦО «Дунай» реалізоване у середовищі інтегрованої розробки Borland Delphi та Microsoft DDK. Система управління базою даних, яка надає змогу функціонування та виконання всіх завдань системи «Дунай» реалізована у СУБД InterBase. Програмне забезпечення КІСЦО «Дунай» детально протестоване і може надійно функціонувати в системах Windows 7 або Windows Server 2005. Застосування цього програмного забезпечення надає системі [1]:

- простий і зручний графічний інтерфейс користувача;
- гнучку реляційну базу даних з можливістю швидкого пошуку і запитів (вибірок);
- вбудовану роботу з картами і планами об'єктів;
- гнучку систему розмежування доступу до функцій меню системи;
- отримання комплекту готових і формування нових звітних форм про роботу пульта централізованого спостереження, додавання і редагування інформації про реальні причини тривоги, формування добової відомості, відомостей для напряму електромонтерів по заявках та інше;
- зберігання бази даних на всіх комп'ютерах, її редагування в реальному режимі часу;
- повну взаємозамінюваність і резервування ПЕОМ за рахунок мережевих можливостей програмного забезпечення і 100%-ного дублювання інформації в реальному режимі часу;
- простий спосіб організації декількох робочих місць на пульті централізованого спостереження великої місткості з паралельною обробкою сповіщень;
- можливість сполучення пультових крайових пристроїв і ПЕОМ за допомогою зовнішнього програмного драйвера (інформація про

специфікацію і архітектуру драйвера, а також приклади можуть бути передані виробникам устаткування).

ПО КІСЦО «Дунай» складається з декількох програм (пакету програм): автоматизованого робочого місця оператора (програма «Дунай»), програми огляду журналів подій, створення звітів і відомостей (програма «Журнали і Звіти»), програми «Резервного копіювання БД». Кожна з програм виконує свої специфічні функції.

Для надійного функціонування програмного забезпечення системи «Дунай» на платформі операційної системи Windows 7 бажано мати комп'ютери, що забезпечені наступними технічними характеристиками: процесор Intel Core i3 і вище, оперативна пам'ять 2 Гбайт, SVGA відеоадаптер з дозволом 1280x1024 і від 256 Мб ОЗУ, мережеву плату із швидкістю передачі даних 100 Мбайт/сек., жорсткий диск місткістю від 350 Гбайт, декілька портів послідовного обміну (COM-портів) в стандарті RS-232 (кількість портів залежить від планованої кількості мультиплексорів), клавіатура, маніпулятор, звукова плата і акустична система, DVD-ROM привід з будь-якою швидкістю читання даних. Додаткове устаткування: лазерний або струменевий принтер. При виборі конфігурації комп'ютера слід враховувати кількість об'єктів, що охороняються.

2 ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА ФУНКЦІОНУВАННЯ ІНТЕГРОВАНОЇ ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ ПРОТИПОЖЕЖНОГО НАГЛЯДУ

2.1 Визначення функціональних можливостей підсистеми протипожежного нагляду

Програмне забезпечення КІСЦО «Дунай» працює в локальній мережі по протоколу TCP/IP. При цьому всі потоки інформації від об'єктового устаткування поступають на один комп'ютер – «майстер». Копії програми, запущені на інших машинах, працюють у режимі клієнтів. Для оператора частини охорони МНС вони володіють однаковою функціональністю з «майстром». Протокол зв'язку майстер-комп'ютера і клієнтів надійно функціонує в локальній мережі, що забезпечує достатньо велику швидкість передачі інформації. При використанні в локальній мережі виправдано те, що при підключенні будь-якого клієнта база даних переписується на нього з «майстра», що забезпечує резервування інформації і спрощує роботу програми «Дунай», оскільки, незалежно від того де вона запущена – на «майстрі», або на клієнті, робота з базою даних здійснюється однаково: в мережі розповсюджуються тільки внесені зміни. Але встановити програму «Дунай» на комп'ютері, пов'язаному з «майстром» модемним з'єднанням виявляється проблематичним. Для підключення таких клієнтів програмне забезпечення «Дунай» підтримує спеціальний протокол, призначений для зв'язку через GSM-модем. Можливості віддаленого клієнта більш обмежені, але такий клієнт виконує функції віддаленого робочого місця спостереження за пожежними тривогами і не припускає внесення змін в базу даних або управління устаткуванням [1, 2].

Пульт віддаленого спостереження в автоматичному режимі одержує від програми «Дунай» сповіщення про тривоги, що виникають на будь-яких об'єктах, на яких встановлено пожежне устаткування, підключене до КІСЦО «Дунай». Програмне забезпечення КІСЦО «Дунай» має функціональну можливість здійснення розподілу охоронних тривог від пожежних сповіщень. При цьому сигнали від об'єктового устаткування приходять по призначених для цього каналах зв'язку (зайнята або виділена телефонна лінія, виділена радіо частота, пакетний радіозв'язок) на пульт централізованого спостереження системи «Дунай».

Здійснив аналіз сигналів, які поступають від датчиків спостереження, програма «Дунай» формує спеціальне сповіщення, що посилається на пульт спостереження для його обробки оператором.

Підтримка віддаленого робочого місця оператора пожежної частини та реалізація інтегрованої інформаційної підсистеми протипожежного нагляду є подальшим розвитком КІСЦО «Дунай» у напрямі відкритості системи [4].

Введення такої підсистеми забезпечить можливість розширювати систему, додаючи в її склад нового апаратного та програмного забезпечення, що виконує нові функції і що взаємодіє з вже існуючим програмним забезпеченням і устаткуванням через стандартні протоколи (рис. 2.1).

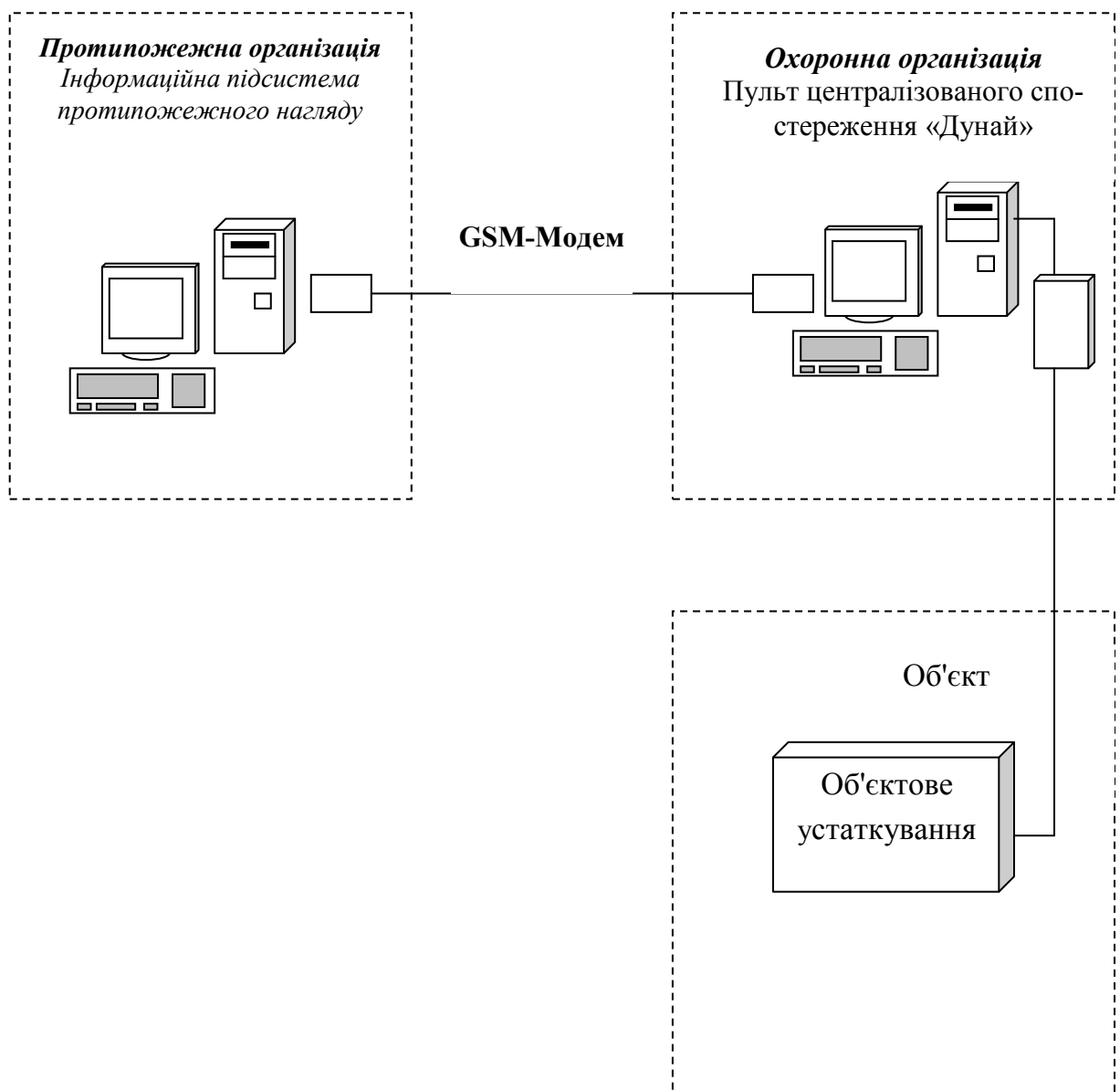


Рисунок 2.1 – Віддалене робоче місце оператора підсистеми протипожежного нагляду в структурі КІСЦО «Дунай»

Програмне забезпечення інтегрованої інформаційної підсистеми протипожежного нагляду повинно відповідати двом основним вимогам:

- здійснювати взаємодію з пультом централізованого спостереження КІСЦО «Дунай» по протоколу, що визначає формат передачі повідомлень про тривоги і забезпечити оператору (черговому) можливість максимально оперативно і безпомилково реагувати на сповіщення;
- забезпечити оперативний доступ до інформації, що до дій та стану об'єкта спостереження.

2.2 Вибір та обґрунтування архітектури підсистеми

Найбільш ефективну роботу з централізованої базою даних (БД) забезпечує архітектура клієнт-сервер. На відміну від системи віддаленої обробки, в якій є тільки один комп'ютер, клієнт-серверна система складається з безлічі комп'ютерів, об'єднаних в мережу. Комп'ютери звані клієнтами, займаються обробкою прикладних програм. Комп'ютери, звані серверами, займаються обробкою БД [4].

Тип комп'ютерів, що використовуються в якості клієнтів може бути різним, це можуть бути великі ПЕОМ або мікрокомп'ютери.

Однак, як правило, функції клієнтів виконують майже завжди ПК. У ролі сервера може виступати комп'ютер будь-якого типу, але з економічних причин функції сервера найчастіше також виконують ПК, але мають більш високу продуктивність [5].

На сервері мережі розміщується БД і встановлюється потужна серверна СУБД – сервер баз даних. Сервер БД – це програмний компонент, що забезпечує зберігання великих обсягів інформації, її обробку та подання необхідної інформації користувачам в мережевому режимі.

На комп'ютері-клієнті додаток-клієнт формує запит до БД. Серверна СУБД забезпечує інтерпретацію запиту, його виконання, формування результату запиту і пересилання його по мережі на клієнтський комп'ютер.

Клієнт інтерпретує програму необхідним чином і видає її користувачу.

Клієнтська програма може також посилати запит на оновлення БД і серверна СУБД внесе необхідні зміни в БД. Схема архітектури клієнт-сервер наведена на рис. 2.2.

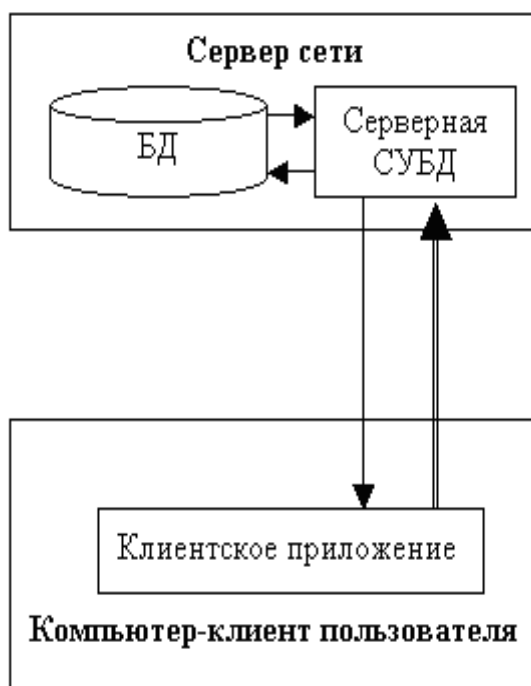


Рисунок 2.2 – Схема функціонування архітектури «клієнт-сервер»

В архітектурі клієнт-сервер функції клієнтського додатка і серверної СУБД розділені. Функції клієнтської програми розбиваються на наступні групи [6]:

- введення-виведення даних (презентаційна логіка) – це частина коду клієнтської програми, яка визначає, що користувач бачить на екрані, коли працює з додатком;
- бізнес-логіка – це частина коду клієнтської програми, яка визначає алгоритм вирішення конкретних завдань програми;
- обробка даних всередині програми (логіка бази даних) – це частина коду клієнтської програми, яка пов'язує дані сервера з додатком. Для зв'язку з цим використовується процедурна мова запитів SQL, за допомогою якого здійснюється вибірка і модифікація даних в серверних СУБД.

Сервер баз даних в загальному випадку здійснює цілий комплекс дій по управлінню даними. Основними серед них є такі:

- виконання запитів користувачів на вибір і модифікацію даних і метаданих, одержуваних від клієнтських додатків, що функціонують на ПК локальної мережі;
- зберігання та резервне копіювання даних;

- підтримка посилювальної цілісності даних згідно з визначеними в БД правилами;
- забезпечення авторизованого доступу до даних на основі перевірки прав і привілеїв користувача;
- протоколювання операцій і ведення журналу транзакцій.

При здійсненні клієнт-серверної обробки зменшується мережевий трафік, завдяки здійсненню передачі тільки результатів запитів. Вантаж файлових операцій лягає в основному на сервер, який має більші системні ресурси, ніж у комп'ютерів-клієнтів, і тому здатний швидше обслуговувати запити. Як наслідок цього, зменшується потреба клієнтських додатків в оперативній пам'яті. Оскільки сервери здатні зберігати велику кількість даних, то на комп'ютерах-клієнтах звільняється значний обсяг дискового простору для інших додатків.

Підвищується рівень несуперечності даних і суттєво підвищується ступінь безпеки БД, так як правила цілісності даних визначаються в серверній СУБД і є єдиними для всіх додатків, що використовують цю БД. Є можливість зберігання бізнес-правил (наприклад правил посилювальної цілісності або обмежень на значення даних) на сервері, що дозволяє уникнути дублювання коду в різних клієнтських додатках, що використовують загальну базу даних.

Всі серверні СУБД мають клієнтську частину, яка звертається до БД за допомогою СУБД. Між клієнтським додатком і СУБД не існує прямого зв'язку і додатково вбудовуються програмні модулі, що дозволяють клієнтському застосуванню отримувати доступ до БД, який створюється за допомогою різних СУБД. Такі модулі називаються механізмами доступу до даних. Існує два основних способи доступу до даних з клієнтських додатків: використання прикладного інтерфейсу і використання універсального програмного інтерфейсу [6].

Прикладний програмний інтерфейс (API – Application Programming Interface) являє собою набір функцій, що викликаються з клієнтського застосування. Він може працювати тільки з СУБД даного виробника і при її заміні доведеться переписувати значну частину коду клієнтського додатку. Прикладний програмний інтерфейс різний для різних СУБД.

Універсальний механізм доступу до даних забезпечує можливість використання одного і того ж інтерфейсу для доступу до різних типів СУБД. Зазвичай він реалізований у вигляді спеціальних додаткових модулів, які називаються драйверами [7].

Найбільш поширеним програмним інтерфейсом, що забезпечує доступ до даних конкретної бази даних є ODBC (Open Database Connectivity) фірми Microsoft. В рамках ODBC програмний додаток безпосередньо взаємодіє з диспетчером драйвером, посилаючи йому ODBC-виклики. Диспетчер драйверів відповідає за динамічне завантаження потрібного ODBC-драйвера, через який звертається з сервера баз даних. ODBC-драйвер виконує всі виклики ODBC-функцій і «переводить» їх на мову джерела даних. СУБД зберігає і виводить дані у відповідь на запити з боку ODBC-драйвера.

Завдання ODBC-джерела даних є дією, яка здійснюється засобами операційної системи, що управляє комп'ютером. Призначене для користувача DSN-джерело даних, доступний тільки активному користувачу на поточному комп'ютері. Файловий DSN – джерело даних, які можуть застосовувати спільно різні користувачі, у яких встановлені однакові ODBC-драйвери. Системний DSN – джерело даних, доступне всім користувачам і службам поточного комп'ютера [6].

Застосування архітектури клієнт-сервер дає можливість поступового нарощування інформаційні системи будь-якого призначення. Поділ загальної бази даних на корпоративну і персональні дозволяє зменшити складність проектування баз даних в порівнянні з централізованим варіантом, а значить знизити ймовірність помилок при проектуванні і вартість проектування.

Найважливішою метою застосування бази даних в інформаційних системах є забезпечення незалежності даних від прикладних програм, це дає можливість користувачам не займатися проблемами представлення даних на фізичному рівні: розміщення даних в пам'яті, методів доступу до них.

Така незалежність досягається підтримуваним СУБД багаторівневим поданням даних в базі даних на логічному (призначеному для користувача) і фізичному рівнях. Завдяки СУБД і наявності логічного рівня представлення даних забезпечується відділення концептуальної (понятійної) моделі бази даних від її фізичного представлення в пам'яті ПЕОМ.

Найважливішим параметром великої інформаційної системи є швидкість при значній кількості користувачів, а також надійність, масштабованість і безпеку. Все це забезпечує архітектура "клієнт-сервер". Така архітектура дозволяє оптимально розподілити роботу між клієнтськими і серверної частинами системи: тепер додаток, що працює на робочій станції, не читає записи бази даних "прямо", а посилає запити на сервер, де вони приймаються і послідовно відпрацьовуються спеціальними програмами. В результаті на робочу

станцію надходять тільки оброблені дані, що радикально скорочує інформаційні потоки в обчислювальній мережі.

2.3 Вибір інтегрованого середовища розробки інтерфейсу підсистеми

Lazarus – це IDE (Integrated Development Environment) – інтегроване середовище розробки програм, що використовує компілятор FPC (Free Pascal Compiler), редактори коду, форм, Інспектор Об'єктів, відладчик і багато інших інструментів. Середовище Lazarus – це RAD (Rapid Application Development) – середовище швидкої розробки додатків [8].

Інтегроване середовище розробки програм Lazarus став першою (і поки єдиною) безкоштовною IDE, доступною освітнім і державним установам абсолютно безкоштовно. Більш того, Lazarus є проектом Open Source – проектом з відкритим вихідним кодом. Багато програмістів по всьому світу беруть участь в його розвитку, вихідний код Lazarus доступний для вивчення і модифікації. Lazarus має підтримку безлічі мов, в тому числі і російською та українською, що вигідно відрізняє його від інших IDE.

FPC – кросплатформовий інструмент, що підтримує величезну кількість платформ. Серед них – AmigaOS, DOS, Linux, BSD, OS/2, MacOS і Win32.

Інтегроване середовище розробки (Integrated Development Environment) IDE Lazarus – це вільне середовище розробки програмного забезпечення для компілятора Free Pascal мовою Object Pascal і поширюється під вільними ліцензіями GNU General Public License (GNU GPL) і GNU Lesser General Public License (GNU LGPL). Вона являє собою середовище з графічним інтерфейсом для розробки програм, схожа на Delphi, і базується на кросплатформовій бібліотеці візуальних і не візуальних компонентів LCL (Lazarus Component Library), сумісних з VCL Delphi. Такого набору компонентів Lazarus досить для створення додатків з графічним інтерфейсом, які працюють з базами даних та Інтернетом. В даний час Lazarus працює під управлінням операційних систем (ОС) Linux, Mac OS X, UNIX-подібних, Windows, Android, в результаті чого додатки, створені під одну ОС, без праці можуть бути перекомпільовані під іншу ОС. Тобто, використовуючи середу Lazarus, можна створювати кросплатформові додатки. Крім того середовище Lazarus має досить багато локалізацій, в тому числі російську та українську та надає можливість кроссплатформової розробки додатків в Delphi-подібному оточенні. Дозволяє досить нескладно переносити Delphi-програми з графічним інтерфейсом в різні

операційні системи: Linux, FreeBSD, Mac OS X, Microsoft Windows. Починаючи з Delphi XE2 в самому Delphi є можливість компіляції програм для Mac OS X і OS. Lazarus заснований на бібліотеці візуальних компонентів Lazarus Component Library (LCL) [5, 7].

Lazarus має інтерфейс налагодження (використовується зовнішній відладчик GDB) і простий перехід для Delphi програмістів завдяки близькості LCL до VCL. Пакет має повністю Юнікодний інтерфейс (UTF-8) і редактор, тому відсутні проблеми з імпортом коду, що містить національні символи. У Lazarus є потужний редактор коду, що включає систему підказок, гіпертекстову навігацію по вихідним текстам, а також форматування коду з використанням механізмів Jedi Code Format. У ньому здійснена підтримка двох стилів асемблера: Intel і AT & T (підтримуються з боку компілятора). У Lazarus здійснена підтримка безлічі типів синтаксису Pascal: Object Pascal, Turbo Pascal, Mac Pascal, Delphi, підтримуваних з боку компілятора. Він має власний формат управління пакетами. Недоліками середовища Lazarus є відсутність повної сумісності з Delphi (хоча на відміну від Delphi надає можливість створювати більш кросплатформені додатки), а також відсутність повної документації, хоча вона доступна у вигляді Wiki-підручників, які можуть редагувати самі користувачі [8].

Бази даних займають важливу роль в сучасному світі, їх використовують для створення більшості сайтів, інформаційних систем, різних програм. У світі налічується величезна кількість СУБД (система управління базою даних), як комерційних MsSql, Oracle, Interbase так і безкоштовних начебто MySQL, PostgreSQL, Firebird, Sqlite.

Lazarus дозволяє працювати з більшістю з них, для цього в середовищі необхідно встановити компоненти для роботи з відповідними СУБД. В основі більшості баз даних лежить мова SQL (мова структурованих запитів), яка дозволяє отримувати та обробляти набори даних. Стандартний набір компонентів для роботи з базою даних в Lazarus можливо застосувати з вкладки SQLdb. Даний компонент SQLdb містить набір для роботи з різними СУБД [9].

TIBConnection – компонент для підключення к СУБД Interbase.

TFBEventMonitor – компонент для роботи з подіями Interbase.

TSQLDBLibraryLoader – компонент для завантаження бібліотеки доступу до даних.

TSQLScript – компонент для роботи з великою кількістю SQL коду.

Для підключення до будь-якої СУБД на форму слід розмістити необхідний компонент, що дозволить налаштувати підключення до необхідної бази даних. Так само слід додати на форму компонент TSQLTransaction, який забезпечує цілісність роботи з даними.

2.4 Вибір СУБД

СУБД InterBase – це система управління базами даних, що поставляється корпорацією Borland для побудови додатків з архітектурою клієнт-сервер довільного масштабу. Сервер InterBase – це кросплатформова СУБД, що підтримує більшість операційних систем: Windows, Linux, Unix, Solaris, Mac OS та інші [10].

СУБД InterBase побудована на версійної архітектурі зберігання даних. Цей підхід має ряд переваг перед блокувальними СУБД:

- для відновлення баз даних InterBase після системного збою, немає необхідності підтримки низки транзакцій;
- клієнти, які читають дані ніколи не блокують клієнтів, які здійснюють запис даних.

Сервер InterBase має цілу низку переваг, що вигідно відрізняють його від інших СУБД:

- оновлювані уявлення View;
- двофазне підтвердження транзакцій;
- ефективний механізм тригерів;
- серверна обробка Blob-полів (Blob-filters);
- події (повідомлення);

Шифрування мережевого трафіку, бази даних, а також окремих стовпців БД. Мова InterBase SQL сумісна зі стандартом SQL-92. Крім того, Interbase server підтримує розширення стандартної підмножини мови SQL за рахунок призначених для користувача функцій UDF (User Defined Functions). InterBase SQL надає розширені можливості SQL для збережених процедур і тригерів – PSQL. Основними утилітами сервера Interbase є:

- InterBase Server Manager – програма управління запуском сервера InterBase. Вона дозволяє отримувати інформацію про стан сервера і режими його запуску. Дозволяє змінювати головний каталог, в якому встановлений сервер і інші команди, що відповідають за налаштування сервера.

- IBConsole – консоль сервера InterBase. Призначена для управління сервером InterBase. Дана програма забезпечує: управління локальними і віддаленими серверами; управління БД; інтерактивне виконання SQL-запитів.
- SQL Monitor – програма відстеження порядку виконання SQL-запитів до віддалених БД. Програма представляє собою інструмент, що дозволяє відстежувати виконання SQL-запитів до віддалених БД. Вона відстежує операції доступу до віддалених БД за допомогою драйверів SQL Link. Після запуску програма автоматично відслідковує порядок ведення запитів, ведучи журнал роботи з віддаленої БД.

Для збереження та швидкого доступу до даних інтегрованої інформаційної системи протипожежного нагляду обрана СУБД InterBase. Повнофункціональна СУБД InterBase, забезпечує легку масштабованість, не вимагає складного адміністрування, підтримує Unicode, відповідає стандартам SQL, кросплатформова. Вона підтримує найширший діапазон способів використання – від окремого вбудованого пристрою до складних корпоративних рішень з сотнями користувачів. СУБД InterBase перевірена в таких критично важливих системах, що вимагають надійності і безвідмовності, як рішення для медицини, залізничного транспорту, міжнародних торгових системах, точках роздрібної торгівлі, CRM, службах реагування на надзвичайні ситуації [10, 11].

InterBase виконує шифрування стовпців даних, відстеження змін на рівні полів, вирішує завдання представлення даних для конкретного користувача, проводить журналізацію, здійснює відновлення даних на конкретний момент часу. Всі ці можливості надають легкий, компактний та простий у використанні та адмініструванні засіб для реалізації розподілених систем.

Ця СУБД має мінімальні вимоги до простору на диску і розміру пам'яті, що робить її невимогливою при розгортанні де завгодно – як на серверах з Windows або Linux (на своєму сервері або в хмарі), так і на мобільних пристроях. Кросплатформена дискова архітектура зберігання бази даних InterBase принципово знижує проблеми з розгортанням систем.

При використанні InterBase дуже легко здійснювати переноси даних між базами для розробки, тестування і експлуатації систем – не залежно від того, під якими операційними системами працюють ці БД. InterBase повністю відповідає вимогам ACID (атомарність, узгодженість, ізольованість, довговічність).

InterBase також має можливості ведення журналу, який дає вам як швидкість повної обробки в пам'яті, так і захист даних при безпосередніх операціях запису. Журнал роботи забезпечує відсутність втрат даних. В результаті можливо значно збільшити швидкодію системи і можливість її розширення, уникаючи довгих звернень до диска. Відновлення на конкретний момент часу в InterBase є ще одним прекрасним засобом забезпечення відмовостійкості. Функція журналізації, забезпечує ця можливість перегляду бази даних до потрібного моменту часу в минулому [9, 11].

InterBase сертифікований для використання з Amazon EC2, тому підтримується можливість розгортання InterBase в хмарному середовищі, що дозволяє додаткам отримати миттєвий доступ до критично важливих ресурсів. Крім того, у СУБД є можливість використовувати локальні копії сервера для підвищення безпеки і продуктивності.

InterBase підтримує всі основні мови програмування, включаючи Java, C, C ++, C#, .NET, Delphi, PHP і Ruby. Він тісно інтегрується з C ++ Builder, RAD Studio, Lazarus та Delphi [12]. При розгортанні систем на будь-якій платформі, СУБД InterBase може автоматично розгортатися разом з цим ПО. Якщо для розробки систем використана інтегрована середовище Visual Studio, то працювати з InterBase можливо через ADO.NET або ODBC, що забезпечить вам легкий доступ до даних.

InterBase підтримує роботу з використовуючи кілька мережевих протоколів: TCP/IP, NetNEUI, IPX/SPX, та забезпечує роботу механізму оптимістичного блокування на рівні запису. Це означає, що сервер блокує тільки ті записи, які реально були змінені користувачем, і не блокує всю сторінку даних цілком. Ця особливість ще більше знижує ймовірність конфліктів при багатому користувальницькому режимі.

У порівнянні з багатьма іншими СУБД, InterBase надає дуже ефективний механізм тригерів: кожна таблиця може мати велику кількість тригерів, які виконуються автоматично при вставці, зміні або видаленні кожного окремого запису, до або після цих подій. Багато функції існуючих СУБД були вперше реалізовані в InterBase – це, зокрема, оновлювані уявлення, події (event alerters), багатовимірні масиви і BLOB-поля [13].

Важливою особливістю сервера InterBase є можливість розширення стандартного набору SQL-функцій за допомогою призначених для користувача бібліотек – User Defined Functions, а також механізми обробки BLOB-полів на сервері за допомогою BLOB-фільтрів. InterBase відрізняється значною стійкістю, оскільки спеціально був спроектований для застосування в

Intranet-додатках, додатках для мобільних пристроїв і вбудованих додатках баз даних.

2.5 Протокол зв'язку інформаційної системи протипожежного нагляду

Для виконання з'єднання інтегрованої інформаційної системи протипожежного нагляду з системою «Дунай» необхідно визначити протокол передачі даних. Реалізація протоколу зв'язку з системою «Дунай» означає можливість виконання програмним забезпеченням інтегрованої підсистеми протипожежного нагляду наступних функцій:

- встановлювати з'єднання з програмою «Дунай» засобами GSM мережі та її каналів зв'язку з використанням GSM-модему. З'єднання вважається встановленим після того, як інтегрована інформаційна підсистема протипожежного нагляду перший раз посилає підтвердження на контрольне повідомлення, яке приходить кожну хвилину.
- контролювати наявність з'єднання (прийом повідомлень типу «контроль лінії» від програми «Дунай» і підтвердження їх прийому посилкою повідомлень у відповідь). Контрольне повідомлення складається з тринадцяти байт: FF000000001E<13h>. Підтвердження – з одного байта з кодом 25 (%).
- одержувати повідомлення про пожежну тривогу на об'єкті від пульта централізованого спостереження «Дунай». Повідомлення складається з унікального номеру ППК, що функціонує у складі комплексної інтегрованої системи «Дунай» і номеру шлейфу, підключеного до цього ППК, з якого прийшло сповіщення про тривогу. Формат повідомлення представлений на рис. 2.3.

М	Р	ППК	Група	Шлейф	С	CR
---	---	-----	-------	-------	---	----

Рисунок 2.3 – Формат повідомлення протоколу зв'язку ПЦН «Дунай» з інтегрованою інформаційною підсистемою протипожежного нагляду

М – номер мультиплексора (1 байт)

P – номер ретранслятора (1 байт)

ППК – номер ППК (4 байти)

Група – номер групи шлейфів (2 байти)

Шлейф – номер шлейфу (2 байти)

C – контрольна сума (2 байти)

CR – байт з кодом 13

Всі поля в цьому повідомленні є ASCII записом шістнадцятиричних чисел. Наприклад, номер мультиплексору – 4 (34h).

ППК з номером шістнадцять – 0010 (30h, 30h, 31h, 30h), таким чином старша цифра знаходиться за молодшою адресою.

Одержане повідомлення про тривогу повинно бути підтверджено посилкою однобайтового повідомлення з кодом 25.

Для забезпечення контролю помилок передачі в повідомленні про тривогу, яке передане пультом централізованого спостереження «Дунай», включена контрольна сума, що є сумою числових значень всієї решти полів повідомлення (окрім завершального байта з кодом 13). Інтегрована інформаційна система протипожежного нагляду обчислює контрольну суму наново і, у разі не збігання її з контрольною сумою, яка передана в повідомленні, не посилає підтвердження програмі «Дунай», що примушує її повторити передачу повідомлення про тривогу.

2.6 Вимоги до інформаційного забезпечення підсистеми протипожежного нагляду

Розробка інтегрованої інформаційної підсистеми протипожежного нагляду забезпечить функціонування віддаленого робочого місця оператора пульта протипожежного нагляду, здійснюючого спостереження за пожежною безпекою об'єктів. Ця підсистема інтегрована у КІСЦО «Дунай» та реалізує робоче місце чергового пожежної охорони, на який надходять повідомлення про пожежні тривоги з об'єктів, де встановлена пожежна сигналізація, підключена до обладнання системи «Дунай».

Програмне забезпечення інтегрованої інформаційної підсистеми протипожежного нагляду, призначене для забезпечення надійного і максимально швидкого реагування чергового пожежної структури, яка територіально вилучена від базового пульта охорони, на пожежні тривоги, що надходять з об'єктів.

Це програмне забезпечення, яке є частиною системи «Дунай», призначене для розміщення на віддаленому робочому місці оператора комп'ютерної системи протипожежного моніторингу і відповідає вимогам пульта, які включають в себе зберігання бази даних про об'єкти, включені в базову систему спостереження. Робота розробленого програмного забезпечення з програмою «Дунай» забезпечується стандартним протоколом зв'язку системи «Дунай» з віддаленим робочим місцем оператора.

Повідомлення, які передаються інформаційній підсистемі протипожежного нагляду від КІСЦО «Дунай», мають мінімальний розмір і включають тільки код, що однозначно ідентифікує шлейф сигналізації на об'єкті, тому інформаційне забезпечення інформаційної підсистеми протипожежного нагляду повинне включати всі необхідні дані про об'єкти, які визначає система «Дунай» (ту її частину, яка має відношення до видаленого робочого місця). Це означає, що інформаційна підсистема повинна зберігати дані в реляційній базі даних і уміти швидко виконувати всі передбачені запити [4].

Програмне забезпечення пульта системи «Дунай» [1, 12] призначено для інформаційного і функціонального забезпечення дій користувачів системи. Структура ПО і дружній інтерфейс користувача орієнтовані на те, щоб в системі могли функціонувати різні прилади.

З урахуванням незамкнутої архітектури апаратних засобів і досить універсального підходу в програмному забезпеченні системи «Дунай» для реалізації інтегрованої інформаційної підсистеми протипожежного нагляду були обрані:

- кросплатформова середа розробки інтерфейсу підсистеми Lazarus, за рахунок можливостей якої потрібно здійснити реалізацію інтерфейсу підсистеми. Інтерфейс підсистеми повинен забезпечувати надійне функціонування надати можливість оператору протипожежної частини (черговому) максимально швидко і безпомилково реагувати на сповіщення, що поступили. Багато в чому це досягається наочністю і продуманістю до дрібниць призначеного для користувача інтерфейсу;
- система управління базою даних InterBase, яка дозволить створити гнучку реляційну базу даних з можливістю швидкого пошуку і запитів (вибірок), а також гнучку систему розмежування доступу до функцій меню підсистеми та зберігання, редагування бази в реальному режимі часу;

Функціонування інтегрованої інформаційної підсистеми протипожежного нагляду передбачено на платформі операційної системи Windows 7 та апаратному устаткуванні з наступними технічними характеристиками: процесор не нижче Intel Core i3, оперативна пам'ять – не менш ніж 2 Гбайт, монітор – SVGA відеоадаптер з дозволом 1280x1024, наявною мережевою картою із швидкістю передачі даних 100 Мбайт/сек., жорсткий диск місткістю від 350 Гбайт і вище, декілька портів послідовного обміну (COM-портів) в стандарті RS-232, зручною клавіатурою та маніпулятором, наявною звуковою платою і акустичною системою. При виборі конфігурації комп'ютера слід враховувати кількість об'єктів, що охороняються.

3 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ ПРОТИПОЖЕЖНОГО НАГЛЯДУ

Для здійснення проектування і розробки програмного забезпечення інтегрованої інформаційної підсистеми протипожежного нагляду необхідно забезпечити виконання основних умов: надійно взаємодіяти з пультом централізованого спостереження «Дунай» по протоколу, що визначає формат передачі повідомлень про тривоги і забезпечити оператору (черговому) можливість максимально оперативно і безпомилково реагувати на сповіщення, що приходять, видаючи йому всю необхідну інформацію про об'єкт, з якого поступила тривога.

В якості СУБД, для бази даних інтегрованої інформаційної підсистеми протипожежного нагляду буде використовуватися SQL-сервер InterBase, якій дозволяє зберігати і обробляти великі обсяги інформації в умовах одночасної роботи з БД множини клієнтських додатків.

3.1 Проектування бази даних підсистеми протипожежного нагляду

Оскільки, в першу чергу, йдеться про об'єкти, за якими здійснюється спостереження, головна суть в базі даних – об'єкт. Оскільки об'єкт володіє довільною кількістю шлейфів (за кодом яких здійснюється передача тривог) для ліквідації множинних атрибутів вводиться суть шлейф сигналізації (зона об'єкту), пов'язана з об'єктом відношенням багато до одного. Одна або декілька обслуговуючих організацій працюють з устаткуванням на всіх об'єктах. Сутність Обслуговуюча організація, пов'язана з об'єктами відношенням один до багатьох. У окрему суть виділені атрибути вулиця, галузь (до якої належить об'єкт), тип устаткування, яку встановлене на об'єкті. Ця сутність визначена постановкою завдання: у базі даних пульта, встановленого в деякому районі міста можуть бути введені всі вулиці району, незалежно від того знаходяться на них об'єкти, на яких здійснюється спостереження чи ні. Приналежність об'єкту до тієї або іншої галузі може впливати на умови надання послуг, і список галузей повинен бути складений наперед.

Сутність Галузь визначається наступними перерахованими атрибутами. Назва об'єкту ідентифікує об'єкт, яка і є ключовим атрибутом для нього («Міська лікарня №1», «Ліцей «Амадеус»). Об'єкт характеризується галуззю до якої він належить («Медицина», «Торгівля»). Адреса об'єкту – це важливий атрибут, що включається в повідомлення про тривогу, яке видається ба-

зою даних і відображається у формі вікна підсистеми протипожежного нагляду. Адреса об'єкта складається з: назви вулиці, номеру будинку, номеру під'їзду, поверху, номера приміщення. Ці поля заповнюються у разі потреби (наприклад, поверх, під'їзд, номер приміщення можуть бути не обов'язкові).

По будь-якому об'єкту нагляду зберігається інформація про організації або власників цього об'єкту, це наступні атрибути: назва організації-власника, керівник організації чи підприємства, чи особа власник приміщень, його службовий телефон, мобільний телефон, домашній телефон; заступник керівника, його службовий телефон, мобільний телефон, домашній телефон; Телефон чергового (якщо такий є) в організації-власника об'єкту. Цей телефон використовується перш за все для сповіщення власника про тривогу на його об'єкті. Це може бути, наприклад, телефон вахтера, що знаходиться в тій же будівлі, що і об'єкт, з якого поступила тривога.

Для гнучкішого уявлення інформації про об'єкт, який може бути державною, комунальною або приватною власністю вводиться поле орендодавця. Наприклад, як орендодавець об'єкту «Міська лікарня №1» виступає Міністерство охорони здоров'я. Організація-орендодавець характеризується наступними атрибутами: назва організації, керівник організації орендодавця, його службовий телефон, мобільний телефон, домашній телефон. Додаткова інформація – про заступника керівника. Також передбачена можливість зберігати його робочий телефон, мобільний телефон, номер домашнього телефону. Потрібно відзначити, що вся ця інформація не обов'язково повинна бути присутньою в базі даних. Наприклад, якщо якісь питання розв'язуються тільки із заступником, не потрібно вводити дані по керівнику, тим більше, включаючи його домашній телефон. У полі керівник або заступник взагалі можна внести іншу відповідальну особу, що відповідає за контакти організації власника (орендодавця) з пожежною охороною. Черговий телефон орендодавця може використовуватися для тих же цілей, що і черговий телефон організації – власника.

Наступні два важливі сутності – Обслуговуюча організація і Устаткування, яке встановлене на об'єкті. Атрибутами сутності Устаткування є інформація про встановлене на об'єкті устаткування – це тип обладнання (прибори), до якого підключені датчики сигналізації. Обслуговуюча організація – організація, що встановила це устаткування і займається його обслуговуванням. Сутність обслуговуюча організація повинна зберігати наступну інформацію: керівник; робочий, мобільний і домашній телефони керівника. Замість імені і координат керівника обслуговуючої організації, в цих полях можуть

зберігатися дані про іншого співробітника обслуговуючої організації, з яким підтримуються контакти.

Сигналізація на об'єкті може охоплювати багато приміщень, тому про тривогу потрібно ідентифікувати по зонах об'єкту (наприклад, складське приміщення №2, вестибюль і т.п.). Звичайно сповіщувачі, контролюючі окреме приміщення (або невеликі суміжні приміщення) підключені до одного шлейфу.

Сигнал тривоги, проходячи через все проміжне устаткування, приходить на ПЦН з якогось конкретного шлейфу об'єктного устаткування. Таким чином, на кожному об'єкті є окремі контрольовані шлейфами ділянки – зони. Кожен шлейф має код і найменування (наприклад, «Пожарна сигналізація в кімнаті №3»), яке є описом місця розташування.

Код шлейфу грає важливу роль в протоколі обміну інформацією з програмою «Дунай» і має формат «М.Р.ППК.Група.Шлейф», де М – номер мультиплексора, Р – номер ретранслятора, ППК – ППК, Група – група шлейфів, підключених до ППК, Шлейф – номер шлейфу.

Хоча атрибути назва б'єкту, вулиця, галузь, устаткування є природними ключовими атрибутами відповідних відносин, для них як для первинних ключів вводиться числовий ідентифікатор. Це скорочує розмір зовнішніх ключів в підлеглих відносинах.

3.1.1 Опис таблиць бази даних і зв'язків між ними

Виходячи з вимог до інформаційної підсистеми, а також з вимог до інформації, що зберігається, було отримано ряд таблиць реляційної бази даних.

На підставі врахування всіх вимог до підсистеми створені таблиці бази даних, які мають наступні поля (табл. 3.1 – 3.7).

Таблиця 3.1 – Сутність «Галузі» – (Branche)

Поле	Тип	Розмір	Опис
Code	Short		Унікальний ID. Первинний ключ
Name	Alpha	100	Назва галузі

Таблиця 3.2 – Сутність «Устаткування» – (Equipment)

Поле	Тип	Розмір	Опис
Code	Short		Унікальний ID. Первинний ключ
Name	Alpha	250	Назва устаткування

Таблиця 3.3 – Сутність «Об’єкти» – (Object)

Поле	Тип	Розмір	Опис
Code	Autoincr		Унікальний ID, первинний ключ
Full_Name	Alpha	255	Назва об'єкту
Branch	Short		Галузь (-> Branch.Code)
A_Street	Short		Вулиця (-> Street.Code)
A_Home	Alpha	20	Будинок
A_Apartment	Alpha	10	Приміщення
A_Doorway	Alpha	10	Під'їзд
A_Floor	Alpha	10	Поверх
Owner	Alpha	200	Власник
Chif_Name	Alpha	100	Ім'я керівника організації-власника
Chif_PhoneW	Alpha	20	Робочий телефон
Chif_PhoneH	Alpha	20	Домашній телефон
Chif_PhoneM	Alpha	20	Мобільний телефон
Depu_Name	Alpha	100	Ім'я заступника організації-власника
Depu_PhoneW	Alpha	20	Робочий телефон
Depu_PhoneM	Alpha	20	Мобільний телефон
Duty_Phone	Alpha	20	Телефон чергового орг.-власника
Rentor	Alpha	255	Орендодавець
RStreet	Short		Вулиця орендодавця
RHome	Alpha	10	Будинок орендодавця
RApartment	Alpha	10	Приміщення орендодавця
RChif_Name	Alpha	100	Ім'я керівника орг.-орендодавця
RChif_PhoneW	Alpha	20	Робочий телефон
RChif_PhoneH	Alpha	20	Домашній телефон
RChif_PhoneM	Alpha	20	Мобільний телефон
RDepu_Name	Alpha	100	Ім'я заступника орг.-орендодавця
RDepu_PhoneW	Alpha	20	Робочий телефон
RDepu_PhoneH	Alpha	20	Домашній телефон
RDepu_PhoneM	Alpha	20	Мобільний телефон
Rduty_Phone	Alpha	20	Телефон чергового орг.-орендодавця
Additional	Memo	240	Додаткова інформація
Equipment	Short		Устаткування
Servises	Short		Обслуговуюча організація

Таблиця Object індексована по полю Full_Name – як наслідок, об'єкти відображаються в алфавітному порядку .

Таблиця 3.4 – Сутність «Вулиця» – (Street)

Поле	Тип	Розмір	Опис
Code	Short		Унікальний ID. Первинний ключ
Name	Alpha	100	Назва вулиці

Таблиця 3.4 – Сутність «Обслуговуюча організація» – (Servises)

Поле	Тип	Розмір	Опис
Code	Short		Унікальний ID. Первинний ключ
Name	Alpha	255	Назва обслуговуючої організації
Chif_Name	Alpha	100	Ім'я керівника організації
Chif_PhoneW	Alpha	20	Робочий телефон
Chif_PhoneH	Alpha	20	Домашній телефон
Chif_PhoneM	Alpha	20	Мобільний телефон

Таблиця 3.5 – Сутність «Шлейфи (Зони)» – (Zones)

Поле	Тип	Розмір	Опис
Code	Alpha	14	Код шлейфу. Первинний ключ
Name	Alpha	254	Назва шлейфу
Code_Obj	Short		Зовнішній ключ об'єкту
Location	Memo	240	Додаткова інформація

Під час надходження тривоги, інтерфейс інформаційної підсистеми протипожежного нагляду відображає інформацію про тривогу у вікні спостереження, але, окрім цього, деь повинна бути зроблена запис про надходження тривоги з часом її надходження.

Якщо у вікні спостереження були не оброблені тривоги, або робота підсистеми була припинена аварійно, при наступному запуску програми ці тривоги повинні знову бути показані оператору.

Окрім тривог підсистема фіксує наступні події: «програма запущена користувачем», «вихід з програми», «установка з'єднання з ПО «Дунай», «з'єднання розірвано користувачем», «аварійний розрив з'єднання».

Для збереження інформації про всі подій перелічені вище, у інтерфейсі підсистеми в наочній області визначена ще одна сутність – Подія, що відбулася (Тривога).

Сутність Подія характеризується такими атрибутами: час, коли подія відбулася; об'єкт, шлейф і його розташування (якщо подія – тривога, що поступила); чи вимагає до тепер увагу оператора (оброблена чи ні). Цю сутність реалізує таблиця Alerts (табл. 3.6).

Таблиця 3.6 – Сутність «Тривоги» – (Alerts)

Поле	Тип	Розмір	Опис
Kod	+Autoincr	14	Унікальний ID. Первинний ключ
Object	Alpha	100	Тривога або подія
Location	Alpha	240	Розміщення шлейфу
Zone	Alpha	20	Шлейф
DatTime	Timestamp		Час виникнення
Obrabotana	Alpha	1	Оброблена чи ні ('y'/'n')

Вона не зв'язана відношенням багато до одного з таблицею Zones і Object. У цих сутностях зберігається інформація про об'єкти, за якими здійснюються спостереження віддаленим пультом спостереження, засобами інтегрованої інформаційної підсистеми протипожежного нагляду зараз.

Якщо спостереження за об'єктом більше не ведеться – він віддаляється з бази даних. Інформація, яка зберігається у таблиці Alerts служить журналом і повинна зберігати інформацію про тривоги достатньо довго (півроку, декілька років).

Об'єкт за цей час може бути просто видалений з бази даних (наприклад, по причині припинення договору надання охоронних послуг). Тому назва об'єкту і шлейфу повинні бути копійовані у відповідні поля таблиці Alerts.

Розрахований на багато користувачів режим доступу до інформаційної підсистеми протипожежного нагляду зумовив використання у базі даних системи Сутності Access – «Права користувачів».

Ця сутність забезпечує розмежування прав доступу та повноважень осіб, що працюють з базою даних засобами інформаційної підсистеми протипожежного нагляду.

Ідентифікація користувача у підсистемі необхідна для занесення в журнал дій оператора, для контролю можливих його помилок при роботі з програмою.

Крім того, розмежування прав доступу в системі гарантує захист даних від навмисної спроби дістати несанкціонований доступ до них (табл. 3.7).

Таблиця 3.7 – Сутність «Права доступу користувачів» – (Access)

Поле	Тип	Розмір	Опис
UserName	Alpha	100	Ім'я користувача
Password	Alpha	10	Пароль
Root	Logical		Ознака користувача-адміністратора
Right1	Alpha	30	База даних «read», «write»
Right2	Alpha	30	Журнал (Alerts) «read», «write»
Right3	Alpha	30	Розрив з'єднання «yes»
Right4	Alpha	30	Зарезервовано
Right5	Alpha	30	Зарезервовано
Setting	Alpha	255	Налаштування інтерфейсу користувача

У реальному проектуванні структури бази даних застосовується метод – так зване, семантичне моделювання. Семантичне моделювання є моделювання структури даних, спираючись на зміст цих даних.

Як інструмент семантичного моделювання використовуються різні варіанти діаграм сутність-зв'язок (ER – Entity-Relationship). Всі варіанти діаграм сутність-зв'язок виходять з однієї ідеї – малюнок завжди наочніше текстового опису. Всі такі діаграми використовують графічне зображення сутностей предметної області, їх властивостей (атрибутів), і взаємозв'язків між сутностями [6].

Ключ сутності – це ненадлишковий набір атрибутів, значення яких у сукупності є унікальними для кожного екземпляра сутності. Ненадлишковість полягає в тому, що видалення будь-якого атрибута з ключа порушується його унікальність. Сутність може мати кілька різних ключів.

Зв'язок – це деяка асоціація між двома сутностями. Одна сутність може бути пов'язана з іншою сутністю або сама з собою. Зв'язки дозволяють по одній сутності знаходити інші сутності, пов'язані з нею.

При розробці ER-моделей треба отримати наступну інформацію про предметну область: список сутностей предметної області; список атрибутів сутностей; опис взаємозв'язків між сутностями.

ER-діаграми зручні тим, що процес виділення сутностей, атрибутів і зв'язків є ітераційним.

Діаграма «Сутність – Зв'язок» дозволяє використовувати наочні графічні позначення для моделювання сутностей та їх взаємозв'язків [7].

Схема сутностей бази даних інтегрованої інформаційної підсистеми протипожежного нагляду і зв'язків між ними, представлені на рис. 3.1.

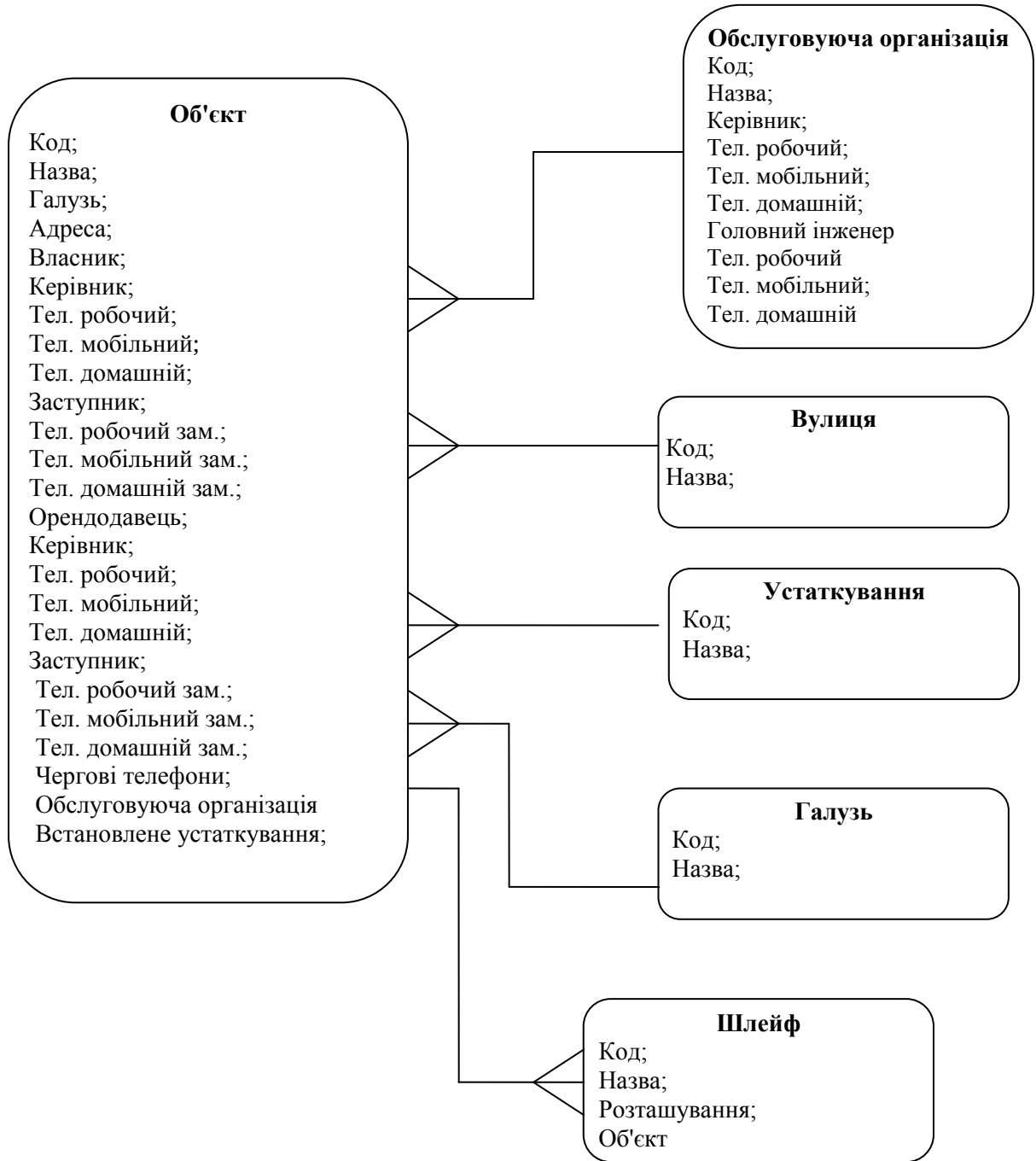


Рисунок 3.1 – Схема бази даних інформаційної підсистеми протипожежного нагляду (діаграма «Сутність – Зв'язок»)

3.2 Проектування функціональних можливостей користувачів підсистеми і інтерфейсу

Головне завдання інтегрованої інформаційної підсистеми протипожежного нагляду – надання зручного та функціонального графічного інтерфейсу для роботи оператора пульта централізованого спостереження за протипо-

жежним станом об'єктів, що охороняються. Основні фактори, за допомогою яких можна оцінити або навіть виміряти зручність використання системи, такі. Адекватність призначеного для користувача інтерфейсу інформаційної підсистеми – це його відповідність тим завданням, які користувачі повинні і хотіли б вирішувати з його допомогою.

Інтерфейс підсистеми повинен сприяти запобіганню та подоланню можливих помилок користувачів. Цей показник тим краще, чим рідше користувачі помиляються при роботі з даним інтерфейсом і чим менше часу і зусиль потрібно для подолання наслідків вже зроблених помилок. Велике значення має також передбачити ризик, пов'язаний з виникненням помилки.

Крім того робота з підсистемою повинна бути настільки зрозумілою, щоб користувач, який вважається фахівцем в системах нагляду за безпекою об'єктів мав змогу з мінімальними витратами часу на навчання роботі з системою легко виконувати всі необхідні дії. Підсистема не повинна перешкоджати ефективній роботі досвідчених користувачів, які працюють з нею довгий час. Інтерфейс підсистеми повинен забезпечувати зручність використання, та звертаючи увагу на спеціальне призначення даного програмного засобу, Одними з основних деталей, що впливають на враження користувача, не був переповнений графічними об'єктами та зайвою інформацією. Інтерфейс також повинен бути функціональним. По його вигляду і структурі користувач повинен одержати уявлення про всі функції підтримуваних програмою.

До додаткової, але дуже корисної можливості, можна віднести можливість програми повідомляти про тривоги звуковим сповіщенням, оскільки оператор пульта, що несе чергування в нічний час, може просто заснути та несвоєчасно здійснити реагування на тривогу.

В інтегрованій інформаційній підсистемі протипожежного нагляду реалізована система доступу, орієнтована змогу доступу багатьох користувачів. Розрахований на багато користувачів режим служить для розмежування повноважень осіб, що працюють з базою даних через програму інформаційної підсистеми протипожежного нагляду. Це забезпечує змогу занесення в журнал дій оператора, для контролю можливих його помилок при роботі з програмою. Цей режим не призначений для гарантованого захисту даних від ненавмисної спроби дістати несанкціонований доступ до них.

Розрахований на багато користувачів режим служить для захисту бази даних від ненавмисного псування користувачем, оскільки, якщо можливість редагувати, переглядати, видаляти інформацію присутня в інтерфейсі програми і не заблокована, черговий, робота якого достатньо монотонна може

зайнятися «вивченням» інтерфейсу програми і щось ненавмисно видалити. Крім того процедура аутентифікації чергового оператора (і користувача з розширеними правами) призводить до того, що він відповідальніше відноситься до всіх своїх дій при роботі з інформаційною системою.

Реалізація розрахованого на багато користувачів режиму здійснена за допомогою розмежування прав доступу в підсистемі: для кожного користувача в базі даних користувачів зберігається його ім'я і пароль. Ім'я просто ідентифікує кожного користувача. Воно використовується при занесенні в журнал записів про дії користувачів.

При вході користувача в програму вводиться тільки його пароль (тому паролі повинні бути унікальними для кожного користувача і програма стежить за цим). Існує один користувач – адміністратор, який володіє правом редагувати права доступу інших користувачів на перегляд, зміну об'єктів бази даних.

Інформація про події з об'єктів поступає в режимі реального часу, і повинна бути максимально швидко оброблена оператором. При здійсненні програмної реалізації інтегрованої інформаційної підсистеми протипожежного нагляду було виділено ряд вимог, які буди забезпечені. У базі даних потрібно зберігати всю необхідну інформацію, підтримувати її цілісність, несуперечність і актуальність. Так само передбачені різні корисні вибірки з бази даних, що дозволяють користувачу одержувати відповіді на питання, що цікавлять його, оперативно і легко.

Апаратне забезпечення, яке застосоване для робочого місця чергового оператора пульта централізованого спостереження повинно бути перш за все надійним надійним. Для забезпечення захисту персонального комп'ютера від нестабільної напруги в електричній мережі і можливості його роботи при аварійному відключенні живлення необхідно використовувати джерела безперебійного живлення (UPS). Оскільки черговий оператор пульта цілодобово працює з інформацією, яку він отримує з монітора, необхідно приділити увагу правильному вибору цього пристрою. Розмір діагоналі 19 дюймів можна рахувати мінімально допустимим, краще використовувати і більш дюймовий монітор. Однією з вимог, що пред'являються до комп'ютера пульта централізованого спостереження, є можливість його цілодобової роботи без виключень. Сьогодні певну (але не абсолютну) гарантію надійності дає використання комплектуючих відомих фірм, що мають вищу, порівняно з іншими, вартість. Навіть така незначна деталь, як зручна миша і клавіатура може зробити роботу оператора комфортнішою.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНТЕГРОВАНОЇ ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ ПРОТИПОЖЕЖНОГО НАГЛЯДУ

Розроблена інтегрована інформаційна підсистема протипожежного нагляду здійснює автоматизоване приймання пожежних тривог з пульта централізованого спостереження «Дунай». Ця інтегрована підсистема повністю підтримує формат передачі повідомлень від КІСЦО «Дунай» і відповідає всім основним вимогам, що пред'являються до такої спеціалізованої підсистеми. Модульна структура програмного комплексу інформаційної підсистеми протипожежного нагляду дозволяє додавати до нього новий програмний код, який реалізує ті чи інші функціональні можливості без переробки і перекомпіляції вже створених і працюючих програмних модулів.

4.1 Реалізація механізму аутентифікації користувачів в системі

Зрозуміло, що спостереження за безпекою будь-то протипожежною чи охоронною, на об'єктах відбувається впродовж цілої доби або постійно, і, отже, з інформаційною підсистемою протипожежного нагляду будуть працювати декілька чергових користувачів-операторів, обов'язки яких передбачають здійснення реагування на події (повідомлення), що надходять з пульта централізованого спостереження «Дунай». Крім того, водночас інший користувач, який має повноваження адміністратора системи, має можливість редагування бази даних об'єктів.

Слід зазначити, що сама база даних інтегрованої інформаційної підсистеми протипожежного нагляду не захищена криптографічно. Імена користувачів і їх паролі зберігаються в таблиці Access бази даних підсистеми. Реалізація механізму аутентифікації в підсистемі забезпечує крім розмежування прав доступу ще і для підвищення відповідальності операторів при роботі з підсистемою, оскільки основні дії оператора чи адміністратора підсистеми, такі як запуск і вихід з програми, установка з'єднання з ПЦС «Дунай» і його розрив протоколюються у журналі.

Для здійснення редагування самих користувачів та розподілу прав доступу у системі одному з них привласнюється ознака «root». Користувач з цією ознакою може додавати користувачів і змінювати їх паролі.

Здійснюючи запуск додатка інформаційної підсистеми протипожежного нагляду з'являється вікно аутентифікації користувача у системі, в якому користувач вводить свій пароль (паролі кожного користувача унікальні). Як-

що пароль співпадає з тим, що зберігається в базі даних, відкривається головна форма програми, де згідно з розподілом прав доступу до інформації можуть бути недоступні деякі функціональні кнопки системи (рис. 4.1).

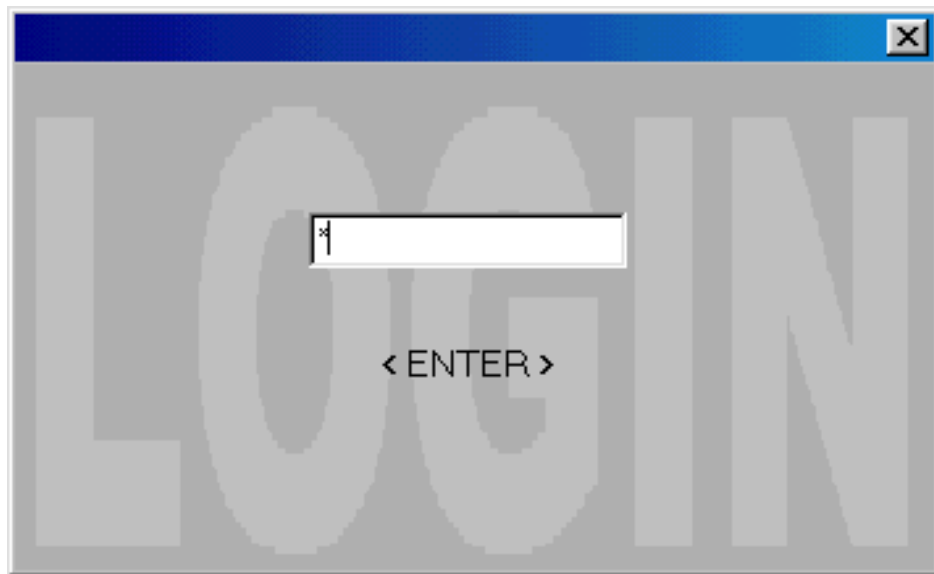


Рисунок 4.1 – Вікно аутентифікації користувачів підсистеми

4.2 Реалізація головної форми інформаційної підсистеми

Інтерфейс інтегрованої інформаційної підсистеми протипожежного нагляду має одновіконну форму, яка реалізує можливість здійснення всіх необхідних функцій обробки пожежних сповіщень користувачем-оператором та здійснення редагування необхідної інформації користувачем-адміністратором. Головна форма розкривається в повноекранному режимі. Більшу частину форми займає активна вкладка, яка перемикається кнопками внизу екрану. Всього є п'ять вкладок: «Тривоги», «Дані», «Пошук», «Журнал», «Налаштування». Залежно від прав користувача, який здійснив успішну ідентифікацію у підсистемі, деякі з них можуть бути недоступні. Кнопки перемикавання закладок, як і інші кнопки, що знаходяться на головній формі, дублюються пунктами головного меню, що дозволяє вибрати їх за допомогою клавіатури.

У верхній частині головної форми знаходяться кнопка «Встановити з'єднання», яка міняється на кнопку «Розірвати з'єднання», якщо зв'язок по GSM каналу з пультом централізованого спостереження «Дунай» в даний момент встановлено. Кнопка «Вихід» реалізує можливість завершення роботи з інформаційною підсистемою. Окрім цього, в центрі верхньої частини

форми, знаходяться два індикатори «Немає зв'язку» і «Тривога», які становляться активними при виникненні цих подій. Підсистема передбачає реалізацію функції звукового сповіщення, якщо стався аварійний розрив зв'язку чи прийшло сповіщення о тривозі. В системі реалізоване у цих випадках запуск файлу Alarm.wav з каталога підсистеми (рис. 4.2).

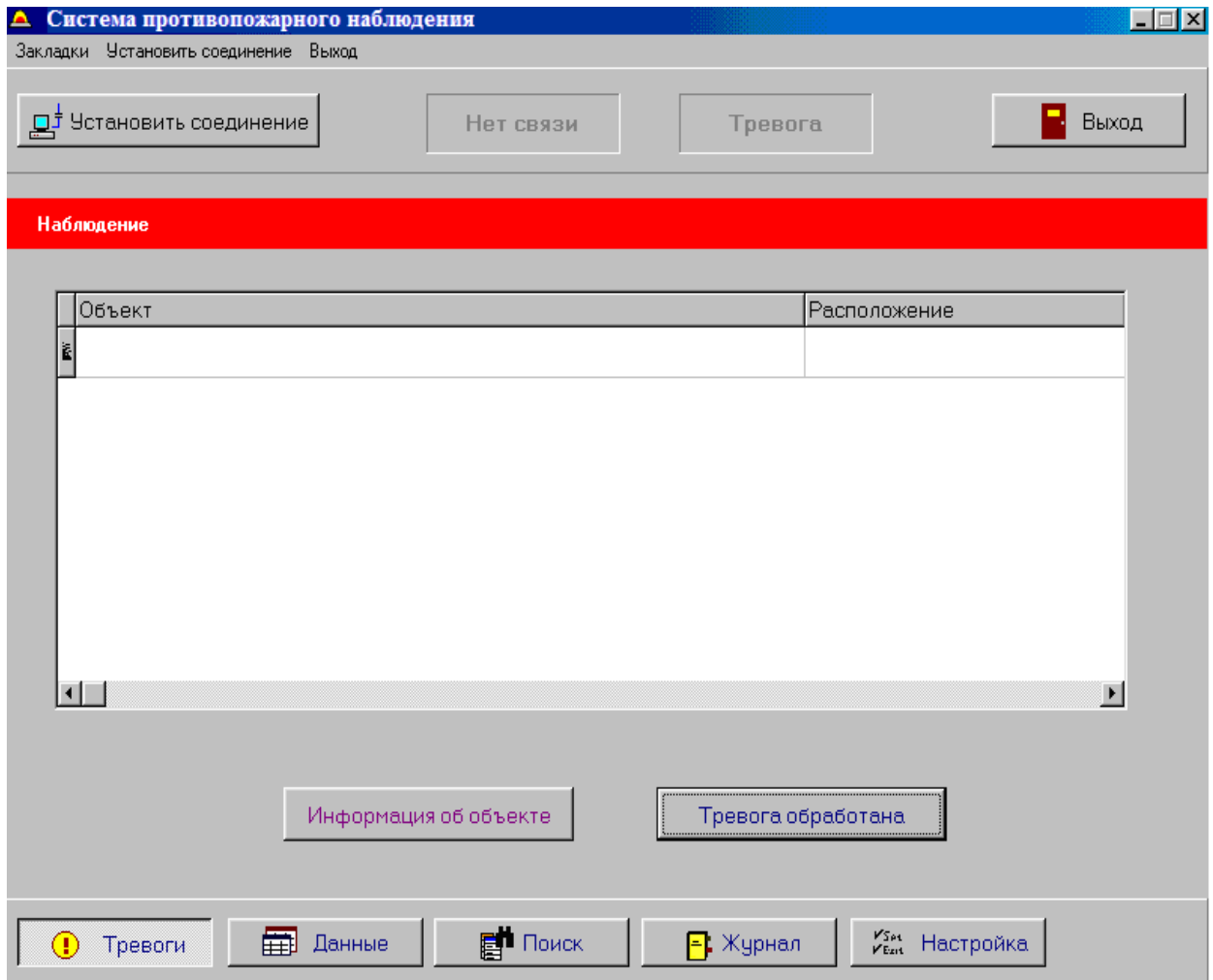


Рисунок 4.2 – Головна форма інтегрованої інформаційної підсистеми протипожежного нагляду

4.3 Реалізація механізму інтеграції підсистеми з ПЦС «Дунай»

Першим кроком, після успішної ідентифікації користувачів у підсистемі, є запуск механізму, що реалізує інтеграцію розробленої підсистеми з ПЦС «Дунай» для успішної передачі пожежних сповіщень. Користувач повинен встановити з'єднання з пультом централізованого спостереження «Дунай»,

щоб на робоче місце чергового користувача-оператора приходили пожежні тривоги з об'єктів.

При натисненні на кнопку «Встановити з'єднання», яка знаходиться на головній формі підсистеми, відкривається вікно в якому показані вільні COM-порти наявні в системі, з яких потрібно вибрати той, до якого підключений GSM-модем (рис. 4.3).

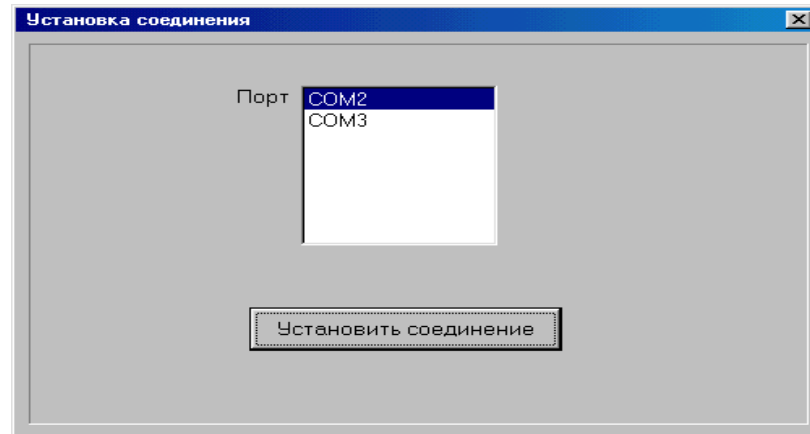


Рисунок. 4.3 – Установка з'єднання підсистеми з ПЦС «Дунай»

Після цього інформаційна підсистема чекає контрольне повідомлення від пульта централізованого спостереження «Дунай» і посилає відповідь на нього. Механізм інтеграції підсистеми також ілюструється відповідним модальним вікном (рис. 4.4).

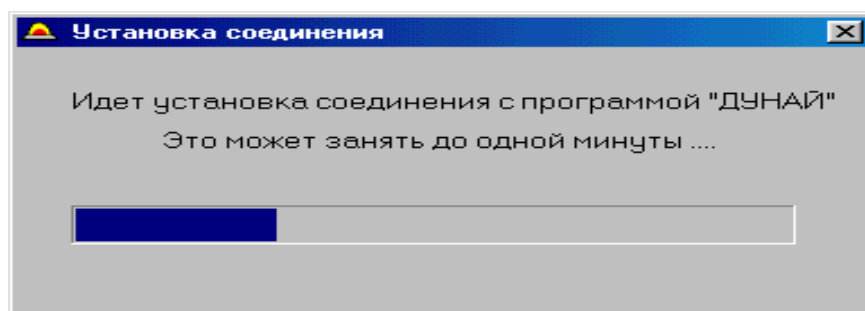


Рисунок. 4.4 – Очікування контрольної посилки від ПЦС «Дунай»

Якщо з'єднання успішно встановлене, кнопка «Встановити з'єднання» змінюється на «Розірвати з'єднання», яка активна, якщо користувач, який здійснює з'єднання, має право в підсистемі здійснювати уручну розрив з'єднання.

4.4 Обробка пожежних тривог у підсистемі

Інтегрована інформаційна підсистема протипожежного нагляду створена для здійснення передачі пожежних тривог з об'єктів ПЦС «Дунай» для надання можливості швидкого реагування протипожежними частинами на виникнення пожег на об'єктах, які знаходяться під наглядом. Коли тривога від ПЦС «Дунай» поступає на робоче місце чергового оператора підсистеми протипожежного нагляду, то підсистема здійснює пошук за кодом шлейфу (зони) в своїй базі даних об'єкт, якому належить цей шлейф.

Якщо об'єкт знайдений, сповіщення додається у вікно тривог на вкладці «Тривоги» і отримання тривоги відображається індикатором на головній формі. Натиснувши на кнопку «Інформація про об'єкт» оператор може переглянути в окремому вікні всю інформацію про об'єкт, на якому відбулася тривога (рис. 4.5).

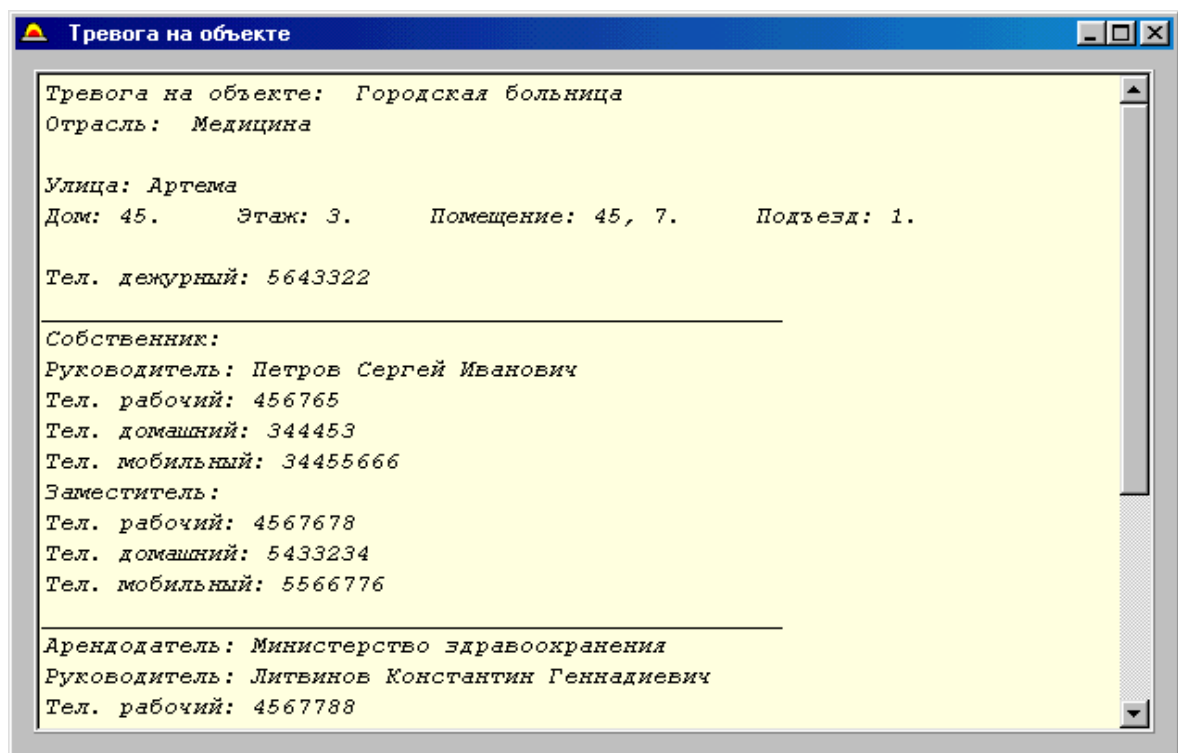


Рисунок. 4.5 – Інформація про об'єкт, по якому надійшла тривога

Коли оператор зробив всі дії по обробці тривоги, він указує на закінчення обробки, натисненням на кнопку «Тривога оброблена». Всі дії оператора з обов'язковим зазначенням часу записуються у журнал, який в змозі переглянути користувач-адміністратор.

4.5 Редагування бази даних

Вкладка «Дані» доступна, якщо користувач, принаймні, має право здійснення перегляду бази даних. На цій вкладці можна обрати, яку з таблиць потрібно редагувати. Після цього, для переходу до редагування потрібного запису таблиці, використовується кнопка «Редагування», для додавання – «Додати», для видалення – «Видалити». Під час внесення змін інші кнопки у вкладці недоступні, а перехід на іншу вкладку викликає вихід з режиму редагування або додавання. Для того, щоб правильно встановлювати зовнішні ключі (наприклад, вулицю для об'єкту) використовуються поля підстановки (рис. 4.6).

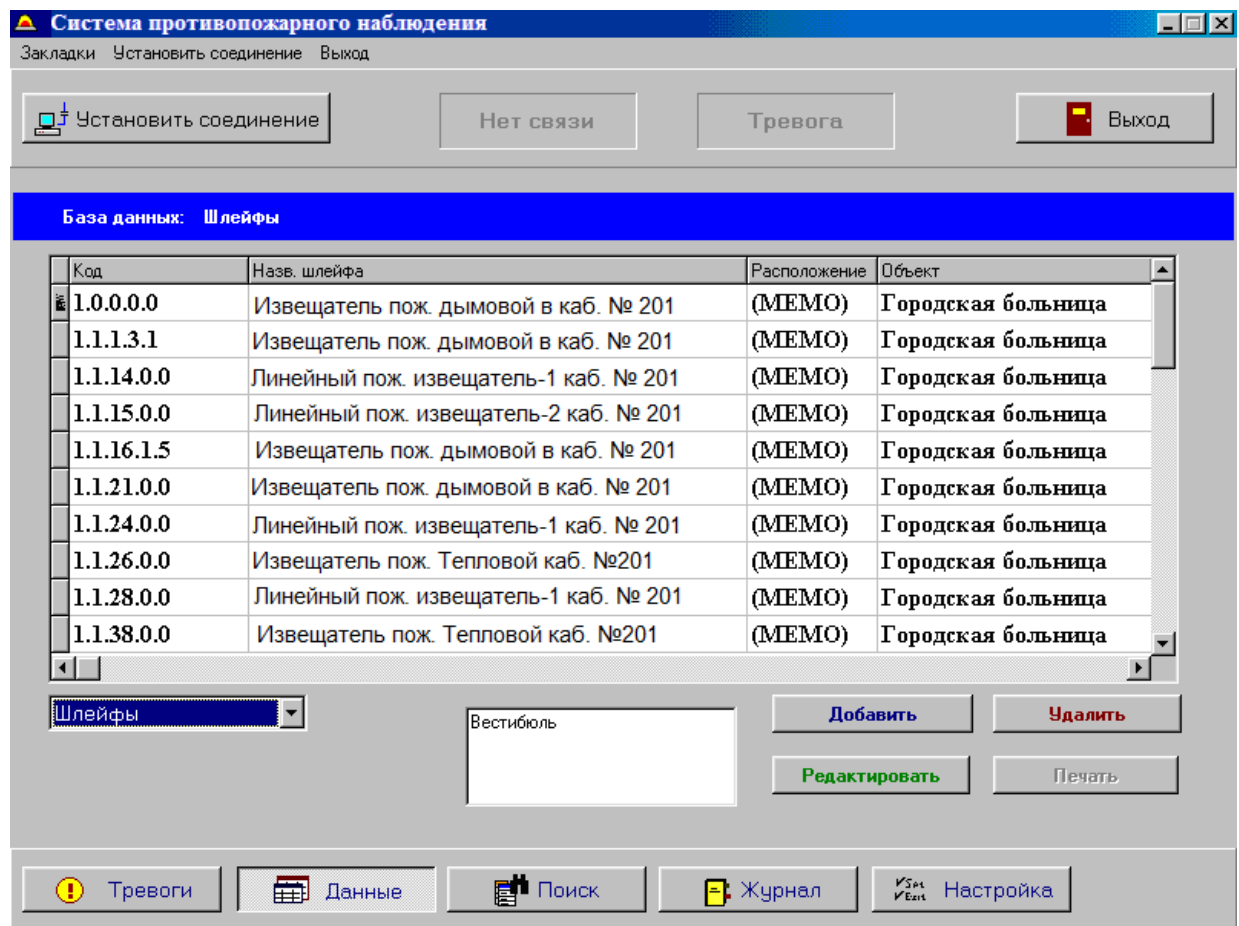


Рисунок. 4.6 – Редагування таблиці «Шлейфи» БД підсистеми

При виборі таблиці «Шлейфи» на вкладку додається багаторядковий редактор для відображення МЕМО-поля додаткової інформації про шлейф.

За допомогою кнопки «Друк» можна роздрукувати дані з різних таблиць. Для таблиці «Об'єкти» обирається, яку саме інформацію потрібно розд-

рукувати: поточний об'єкт, всі об'єкти або об'єкти, що відносяться до конкретної галузі. Разом з об'єктом обираються всі шлейфи сигналізації, встановлені на ньому. Після натиснення на кнопку «Друк» відкривається вікно, в якому можна переглянути, що саме буде роздруковане, встановити розмір шрифту, вибрати принтер і отримати доступ до налаштувань друку Windows.

Якщо права користувача підсистеми обмежені і не передбачають можливості здійснення яких би то дій з базою даних крім перегляду, то користувачу кнопки «Додати», «Редагувати» і «Видалити» будуть недоступні.

4.6 Реалізація механізму пошуку в підсистемі

В інтегрованій інформаційній підсистемі протипожежного нагляду реалізована функція здійснення пошуку в базі даних для всіх таблиць і необхідних полів. Початковою інформацією для пошуку є рядок, для якого перевіряється входження в поля пошуку таблиць без урахування позиції і регістра символів (рис. 4.7).

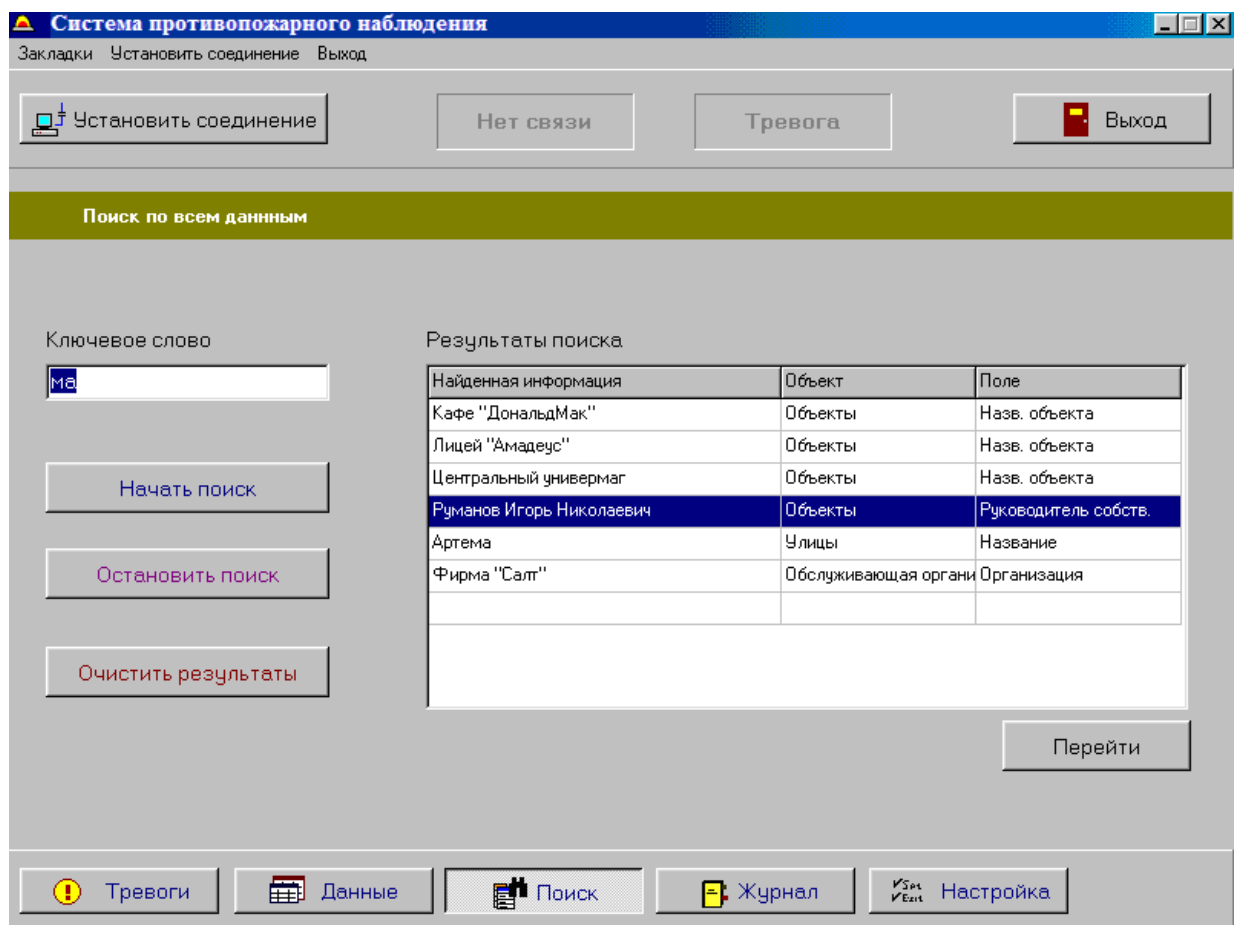


Рисунок. 4.7 – Пошук по базі даних інформаційної підсистеми

Рядок пошуку і його результати відображаються у вкладці «Пошук». У вікні «Результати пошуку» крім значення поля, в якому знайдений підрядок представлена інформація про таблицю і саме поле в якому цей підрядок був знайдений. При виборі рядка в таблиці, що відображена у вікні «Результатів пошуку» і натисненні на кнопку «Перейти» відкриється вкладка «Дані» з тією таблицею, в якій була знайдена інформація. Курсор, в цій таблиці, стоятиме на потрібному рядку і полі. Якщо пошук займає тривалий час, що може бути тільки при дуже великих розмірах бази даних, його можна зупинити за допомогою кнопки «Зупинити пошук».

4.7 Реалізація журналу обліку подій і тривог

Для забезпечення функціональних можливостей реалізації інформаційної підсистеми протипожежного нагляду, була здійснена програмна реалізація журналу обліку подій і тривог у підсистемі. Основним призначенням цього журналу є зберігання з точним зазначення часу (з урахуванням секунд), всіх подій і тривог у системі. Це забезпечить протоколювання як всіх тривог, так і всіх дій користувачів у системі, які забезпечують швидку реакцію та подію (пожарну тривогу, втрата зв'язку і т.п.). У журнал заносяться різні події зафіксовані програмою. Їх можна розділити на кілька категорій:

- тривоги, які надійшли на пульт пожежного спостереження. Як тільки підсистема протипожежного нагляду отримує повідомлення про пожежну тривогу від ПЦС «Дунай», вона видає повідомлення на дисплей і вносить в журнал запис такого виду: «об'єкт, на якому сталася тривога; шлейф – ініціатор тривоги; розташування шлейфа; час (і дата) надходження тривоги;
- втрати і автоматичні відновлення зв'язку з ПЦС «Дунай». Такі події записуються в вигляді записів наступного формату: подія («втрата зв'язку з пультом центрального спостереження» / «відновлення зв'язку з пультом центрального спостереження») та час, в який відбулася подія.
- дії користувачів: «запуск програми одним з користувачів» (користувач ідентифікується по введенню пароля), «вихід з програми». При запуску або закритті інформаційної підсистеми протипожежного нагляду в журналі з'являється запис виду: «користувач «ІМ'Я» запустив програму» / «вихід з програми» та час запуску / виходу.

Здійснивши запуск інтегрованої інформаційної підсистеми протипожежного нагляду, користувач зазвичай встановлює з'єднання з пультом централізованого спостереження «Дунай» (якщо він не збирається, наприклад, просто редагувати базу даних або переглядати якусь статистику по журналу). Підсистема забезпечує спробу встановити з'єднання, і, якщо це вдається, вносить в журнал запис: «встановлено з'єднання з ПЦС «Дунай» та зазначає час, коли подія напустила. Якщо з'єднання встановити не вдалося, підсистема вносить в журнал запис протилежного змісту: «не вдалося встановити з'єднання з ПЦС «Дунай» з зазначенням часу.

При здійсненні програмної реалізація журналу обліку подій та тривоги, реалізована можливість здійснення користувачами підсистеми перегляду у цьому журналі повідомлення окремих груп, за окремий період (наприклад, добу чи місяць), а також друкування обраної інформації (рис. 4.8) .

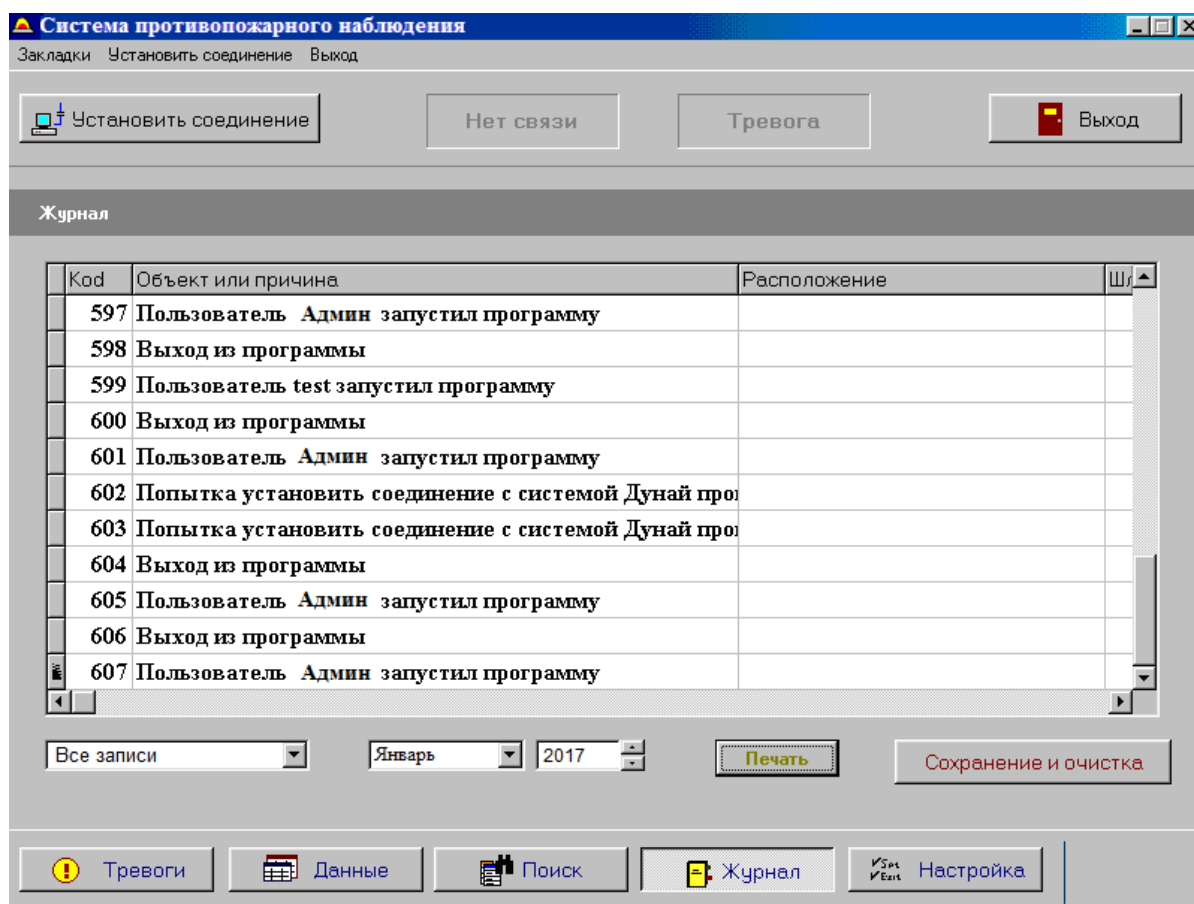


Рисунок. 4.8 – Журнал подій інформаційної підсистеми

Всі тривоги, що поступили з пульта централізованого спостереження «Дунай», і події, типу запуску, виходу, установки з'єднання, зберігаються в базі даних. Відкривши вкладку основної форми програми «Журнал» можна

дістати доступ до цієї інформації. Слід зазначити, що ця вкладка доступна, якщо користувач має, принаймні, право поглядання журналу, якщо ж він має право редагування журналу, на цій вкладці стає доступною кнопка «Збереження і очищення». При виконанні цієї операції викликається програма Backup з підкаталогу Plugins каталогу програми, яка дозволяє зберегти журнал (у тому числі і в архіві), очистити його від застарілої інформації, зберегти всю базу даних.

Інтерфейс інформаційної підсистеми протипожежного нагляду забезпечує можливість здійснення перегляду журналу, обравши потрібну категорію, записи з якої будуть відображені: «всі записи», «тривоги», «розриви чи установки з'єднання», «користувачі». Також можна вибрати за який місяць і рік потрібно проглянути записи (за умовчанням стоїть поточний місяць поточного року). Обрані записи можна роздрукувати, аналогічно інформації з бази даних.

4.8 Налаштування параметрів інтерфейсу підсистеми

Ще одна вимога до інтегрованої інформаційної підсистеми протипожежного нагляду – забезпечення багато-користувальницького режиму. Забезпечення такого режиму у підсистемі забезпечує розмежування повноважень осіб, що працюють з базою даних засобами інформаційної підсистеми протипожежного нагляду за станом об'єктів. Інше його призначення – це занесення в журнал дій оператора, для контролю можливих його помилок при роботі з підсистемою. Цей режим не призначений для гарантованого захисту даних від навмисної спроби отримати несанкціонований доступ до них. Такий захист здійснюється іншими методами (необхідно захищати дані самі по собі, так як скопіювати, видалити, модифікувати файли з даними можна не користуючись програмним забезпеченням віддаленого робочого місця пожежного спостереження).

Розмежування прав доступу до інформації у підсистемі забезпечує захист бази даних від ненавмисного псування користувачами. Якщо можливість редагувати, переглядати, видаляти інформацію присутні в інтерфейсі програми і не заблоковані, черговий оператор, робота якого досить монотонна, може зайнятися «вивченням» інтерфейсу підсистеми і щось ненавмисно видалити. Крім того процедура аутентифікації оператора і користувача з ро-

зширеними правами призводить до того, що він більш відповідально ставиться до всіх своїх дій при роботі з програмою.

Реалізація багато-користувальницького режиму забезпечує у підсистемі зберігання для кожного користувача в базі даних його ім'я і пароль. Ім'я ідентифікує кожного користувача в підсистемі. Воно використовується при занесенні в журнал записів про дії користувачів.

При вході користувача в програму вводиться тільки його пароль (тому паролі повинні бути унікальними для кожного користувача і програма стежить за цим). Існує один користувач – адміністратор, який має право редагувати права доступу інших користувачів на перегляд або зміну об'єктів бази даних.

В інтерфейсі інформаційної підсистеми реалізована вкладенка «Налаштування», яка надає деякі можливості настройки призначеного для користувача інтерфейсу окремо для кожного користувача, і забезпечує можливість здійснення додавання і редагування користувачів адміністратором (рис. 4.9).

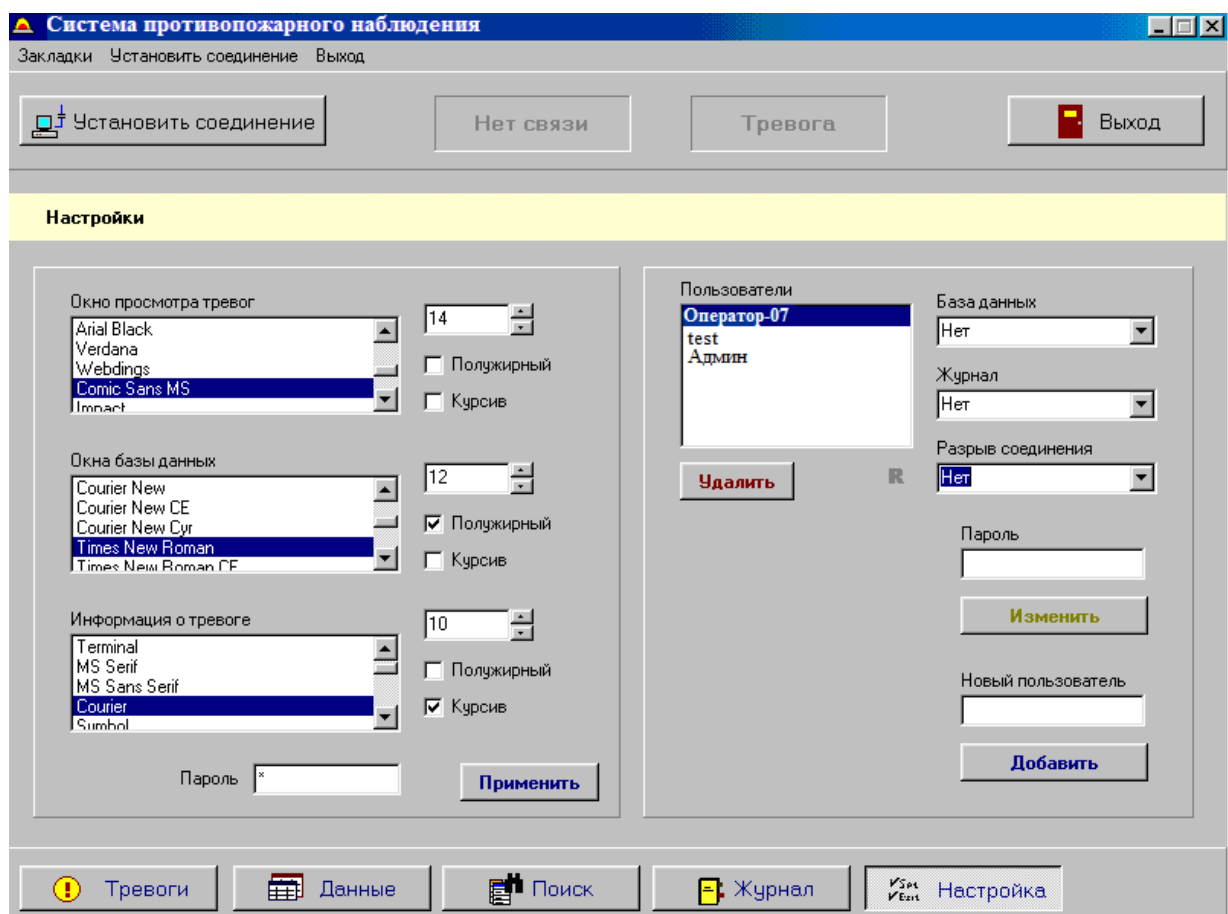


Рисунок. 4.9 – Вікно налаштування параметрів інтерфейсу підсистеми

Вкладка розділена на дві частини – ліву і праву. Ліва частина – наявна та завжди показує доступні опції налаштування інтерфейсу для поточного користувача. Для вкладки «Перегляд тривог», вкладки «Вікно бази даних», і вікна «Перегляд тривог» про об'єкт, на якому відбулася тривога, можна встановити свій шрифт, його розмір і відображення.

Права частина головної форми наявна тільки якщо з програмою працює користувач-адміністратор (у таблиці бази даних «Access» в його записі значення поля root – 'y'). У цій частині реалізована можливість додавання або видалення користувача, крім того реалізована функція зміни паролів користувачів у системі.

Реалізований інтерфейс інтегрованої інформаційної підсистеми протипожежного нагляду забезпечує можливість надання наступних прав.

Для управління базою даних: «зміна / перегляд» – доступна вкладка «Дані» і всі елементи управління на ній; «тільки перегляд» – на вкладці «Дані» доступні кнопки «Додати», «Редагувати», «Видалити»; «Немає» – відкрити вкладку «Дані» неможливо.

Для налаштування журналу подій: «Зміна / перегляд» – доступна вкладка «Журнал» і всі елементи управління на ній; «тільки перегляд» – вкладка «Журнал» доступна, кнопка «Збереження і очищення» – недоступна, «Немає» – відкрити вкладку «Журнал» неможливо.

Для управління з'єднанням з ПЦН «Дунай»: «Розрив з'єднання» – якщо дозволений, то користувач може розірвати з'єднання з ПЦН «Дунай».

Програмна реалізація інтегрованої інформаційної підсистеми протипожежного нагляду, здійснена в рамках магістерської роботи, забезпечує виконання всіх визначених функцій. У складі програмного забезпечення інтегрованої інформаційної підсистеми протипожежного нагляду працюють додаткові програми, які знаходяться у підкаталозі Plugins каталогу підсистеми. Програма prsetup.exe – призначена для настройки принтерів, встановлених в системі. Виклик здійснюється засобами натиснення на кнопку «Налуштування» у вікні друку. Програма backup.exe – зберігає файли бази даних у вказаному каталозі. Вона може також упакувати їх за допомогою архіватора Rar, що також знаходиться в каталозі Plugin. Окрім інтегрована інформаційна підсистема протипожежного нагляду надає можливість зберігання журналу подій у таблицю Alerts.db бази даних у вигляді текстового файлу, і забезпечує виконання функції очищення (видалення) з підсистеми інформації за минулі періоди.

ВИСНОВКИ

В рамках виконання магістерської роботи виконано дослідження вимог та розробка інформаційної підсистеми протипожежного нагляду та реалізація механізму інтеграції підсистеми з комплексною системою централізованого спостереження «Дунай», засобами застосування протоколу, що визначає формат передачі повідомлень про тривоги з цих об'єктів.

Розроблена інтегрована інформаційна підсистема протипожежного нагляду призначена для забезпечення максимально швидкого реагування чергового охоронної структури, яка територіально віддалена від базового пульта охорони на пожежні тривоги, що надходять з об'єктів. Ця інформаційна підсистема інтегрована у програмне забезпечення пульта централізованого спостереження «Дунай», засобами протоколу зв'язку, що визначає формат передачі повідомлень про пожежні тривоги.

Результатом магістерської роботи є програмна реалізація підсистеми, яка призначена для розміщення на віддаленому робочому місці оператора протипожежної системи централізованого спостереження і відповідає вимогам пульта, які включають в себе зберігання бази даних про об'єкти, включених в базову систему спостереження.

Використання реалізованої інформаційної підсистеми надає можливість оператору максимально оперативно і безпомилково реагувати на повідомлення, що приходять, видаючи йому всю необхідну інформацію про об'єкт, з якого поступила тривога.

Інтегрована інформаційна підсистема протипожежного нагляду відповідає всім основним вимогам, що пред'являються до нього. Модульна структура програмного забезпечення дозволяє додавати до нього новий програмний код, який реалізує ті чи інші завдання, без переробки і перекомпіляції вже створених і працюючих програм.

ПЕРЕЛІК ПОСИЛАНЬ

1. Руководство пользователя программного обеспечения комплексной интегрированной системы централизованной охраны «Дунай». – Киев: НВФ «Венбест Л.т.д.» 2007. – 64 с.
2. Каталог разработок научно-внедренческой фирмы «Венбест». – Киев: НВФ «Венбест Л.т.д.», 2010. – 70 с.
3. Прибор приемно-контрольный охранно-пожарный «Дунай». Руководство по эксплуатации. – Киев: НВФ «Венбест», 2000. – 53 с.
4. Бодров В.И., Лазарев Т.Я., Мартыновский Ю.Ф. Математические методы принятия решений. – Тамбов: Изд-во ТГУ, 2004. – 124 с.
5. Подлиновский В.В. Введение в теорию важности критериев в многокритериальных задачах принятия решений. – М.: Физматлит, – 2007, – 64 с.
6. Карпова Т.С. Базы данных: модели, разработка, реализация. – СПб.: Питер, 2001. – 304 с.
7. Грофф Дж. Р., Вайнберг П.Н., Оппель Э. Дж. SQL. Полное руководство. – Издательство «Вильямс», 2015. – 959 с.
8. Алексеев Е. Р., Чеснокова О. В., Кучер Т. В. Free Pascal и Lazarus: Учебник по программированию. – М.: Издательство «ДМК-пресс», 2010. – 442 с.
9. Эрик Эванс. Предметно-ориентированное проектирование (DDD). Структуризация сложных программных систем. – М.: Издательство «Вильямс», 2010. – 444 с.
10. Скляр А. Я. Введение в InterBase. – М.: Издательство «Горячая Линия – Телеком», 2002. – 520 с.
11. Мансуров К.Т. Основы программирования в среде Lazarus. – М.: Издательство «Вильямс», 2010. – 772с.
12. Алексеев Е.Р., Чеснокова О.В., Кучер Т.В. Самоучитель по программированию на Free Pascal и Lazarus. – Донецк.: ДонНТУ, Технопарк ДонНТУ УНИТЕХ, 2009. – 503 с.
13. Ковязин А., Востриков С. Мир InterBase. Архитектура, администрирование и разработка приложений баз данных в InterBase/Firebird/Yaffil. Издание 2-е, дополненное –М.: КУДИЦ-ОБРАЗ, 2002. – 496 с.

Д О Д А Т К И

ДОДАТОК А ОСНОВНІ КОДИ ПРОГРАМНИХ МОДУЛІВ

```

#include <vcl.h>
#include <mmsystem.h>
#pragma hdrstop
#include "Unit1.h"
#include "Unit2.h"
#include "Unit3.h"
#include "Unit4.h"
#include "Unit5.h"
#include "Unit6.h"
#include "Unit7.h"
//-----
#pragma package(smart_init)
#pragma link "CSPIN"
#pragma resource "*.dfm"
TForm1 *Form1;
BOOL ExitProgram;
BOOL ExitWhile;
HANDLE hCom;
OVERLAPPED o;
char buf[14] = {0};
char buf2 = '%';
unsigned long res;
DCB dd;
bool StopFind;

int CALLBACK EnumFontFamProc1( LOGFONT* lpelf, TEXTMETRIC* lpntm, int
FontType, LPARAM lParam)
{
    Form1->ListBox1->Items->Add((char*)(lpelf->lfFaceName));
    Form1->ListBox2->Items->Add((char*)(lpelf->lfFaceName));
    Form1->ListBox3->Items->Add((char*)(lpelf->lfFaceName));
    return 1;
}

void TForm1::SetSetting()
{
    DBGrid2->Font->Name = ListBox1->Items->Strings[ListBox1->ItemIndex];
    DBGrid1->Font->Name = ListBox2->Items->Strings[ListBox2->ItemIndex];
    DBGrid3->Font->Name = ListBox2->Items->Strings[ListBox2->ItemIndex];
    Alert->Memo1->Font->Name = ListBox3->Items->Strings[ListBox3-
>ItemIndex];
    DBGrid2->Font->Size = UpDown1->Position;
    DBGrid1->Font->Size = UpDown2->Position;
    DBGrid3->Font->Size = UpDown2->Position;
    Alert->Memo1->Font->Size = UpDown3->Position;
    DBGrid1->Font->Style = TFontStyles();
    DBGrid3->Font->Style = TFontStyles();
    if (CheckBox3->Checked) {
        DBGrid1->Font->Style = TFontStyles() << fsBold;
        DBGrid3->Font->Style = TFontStyles() << fsBold;
    }
    if (CheckBox4->Checked) {
        DBGrid1->Font->Style = TFontStyles() << fsItalic;
        DBGrid3->Font->Style = TFontStyles() << fsItalic;
    }
}

```

```

    DBGrid2->Font->Style = TFontStyles();
    if (CheckBox1->Checked) DBGrid2->Font->Style = TFontStyles() <<
fsBold;
    if (CheckBox2->Checked) DBGrid2->Font->Style = TFontStyles() << fsI-
tallic;
    Alert->Memo1->Font->Style = TFontStyles();
    if (CheckBox5->Checked) Alert->Memo1->Font->Style = TFontStyles() <<
fsBold;
    if (CheckBox6->Checked) Alert->Memo1->Font->Style = TFontStyles() <<
fsItalic;
}
void TForm1::LoadSetting()
{
    int l,k;
    AnsiString sa[12];
    tAccess->SetKey();
    tAccess->FieldByName("UserName")->AsString = CurrentUser;
    tAccess->GotoKey();
    AnsiString s = tAccess->FieldByName("Setting")->AsString;
    for (int i=0; i<12; i++) {
        l = s.Pos("(");
        k = s.Pos(")");
        sa[i] = s.SubString(l, k - l + 1);
        s.Delete(l, k - l + 1);
        sa[i].Delete(1,1);
        sa[i].Delete(sa[i].Length(), 1);
    }
    for (int i=0; i<ListBox1->Items->Count; i++)
f (ListBox1->Items->Strings[i] == sa[0]) ListBox1->ItemIndex = i;
    for (int i=0; i<ListBox2->Items->Count; i++)
if (ListBox2->Items->Strings[i] == sa[4])ListBox2->ItemIndex = i;
    for (int i=0; i<ListBox3->Items->Count; i++)
if (ListBox3->Items->Strings[i] == sa[8])ListBox3->ItemIndex = i;
    UpDown1->Position = sa[1].ToInt(); UpDown2->Position =
sa[5].ToInt(); UpDown3->Position = sa[9].ToInt();
    if (sa[2] == "yes") CheckBox1->Checked = true; else CheckBox1-
>Checked = false;
    if (sa[3] == "yes") CheckBox2->Checked = true; else CheckBox2-
>Checked = false;
    if (sa[6] == "yes") CheckBox3->Checked = true; else CheckBox3-
>Checked = false;
    if (sa[7] == "yes") CheckBox4->Checked = true; else CheckBox4-
>Checked = false;
    if (sa[10] == "yes") CheckBox5->Checked = true; else CheckBox5-
>Checked = false;
    if (sa[11] == "yes") CheckBox6->Checked = true; else CheckBox6-
>Checked = false;

    Edit4->Text = tAccess->FieldByName("Password")->AsString;
    Edit4Exit(Edit4);
    SetSetting();
}
void TForm1::SaveSetting()
{
    int l,k;
    AnsiString sa[12];
    tAccess->SetKey();
    tAccess->FieldByName("UserName")->AsString = CurrentUser;

```

```

tAccess->GotoKey();
AnsiString s = "";
s += "(" + ListBox1->Items->Strings[ListBox1->ItemIndex] + ")";
s += "(" + Edit5->Text + ") ";
if (CheckBox1->Checked) s += "(yes) "; else s += "(no) ";
if (CheckBox2->Checked) s += "(yes) "; else s += "(no) ";
s += " ";
s += "(" + ListBox2->Items->Strings[ListBox2->ItemIndex] + ")";
s += "(" + Edit6->Text + ") ";
if (CheckBox3->Checked) s += "(yes) "; else s += "(no) ";
if (CheckBox4->Checked) s += "(yes) "; else s += "(no) ";
s += " ";
s += "(" + ListBox3->Items->Strings[ListBox3->ItemIndex] + ")";
s += "(" + Edit7->Text + ") ";
if (CheckBox5->Checked) s += "(yes) "; else s += "(no) ";
if (CheckBox6->Checked) s += "(yes) "; else s += "(no) ";
tAccess->Edit();
tAccess->FieldByName("Setting")->AsString = s;
tAccess->Post();
tAccess->Edit();
tAccess->FieldByName("Password")->AsString = Edit4->Text;
tAccess->Post();
    SetSetting();
}
//-----
__fastcall TForm1::TForm1(TComponent* Owner)
    : TForm(Owner)
{
}
//-----
void __fastcall TForm1::ComboBox1Change(TObject *Sender)
{
    BitBtn15->Enabled = true;
    if (ComboBox1->Text == "Объекты")
        DataSource1->DataSet = tObject;
    if (ComboBox1->Text == "Улицы")
        DataSource1->DataSet = tStreet;
    if (ComboBox1->Text == "Оборудование")
        DataSource1->DataSet = tEquipment;
    if (ComboBox1->Text == "Тип объекта")
        DataSource1->DataSet = tBranche;
    if (ComboBox1->Text == "Обслуживающая организация")
        DataSource1->DataSet = tServices;
    if (ComboBox1->Text == "Шлейфы")
        DataSource1->DataSet = tZones;
    if (DataSource1->DataSet == tZones) {
        Mem1->Show();
        tZones->First();
        BitBtn15->Enabled = false;
    }
    else Mem1->Hide();
    Panel3->Caption="          База данных:  " + ComboBox1->Text + " ";
}
//-----
void __fastcall TForm1::FormShow(TObject *Sender)
{
    HDC hdc = GetDC(Form1->Handle);
    EnumFonts(hdc, 0 , (int(*)())EnumFontFamProc1,0);

```

```

Login->ShowModal();
CurrentUser = tAccess->FieldByName("UserName")->AsString;
LoadSetting();
DBLookupListBox1->KeyValue      =      tAccess->FieldByName("UserName")-
>AsString;
if (!tAccess->FieldByName("Root")->AsBoolean) Panel10->Hide();
Query1->SQL->Text = "INSERT INTO Alert (Object, DatTime, Obrabotana)
VALUES (\\"Пользователь " + tAccess->FieldByName("UserName")->AsString
+ " запустил программу\\", \\" + Now().DateTimeString() + \\", \\"y\\");
Query1->ExecSQL();
Finded1 = new TStringList;
Finded2 = new TStringList;
DbPanel->Left = 1;
DbPanel->Top = 80;
SettingPanel->Left = 1;
SettingPanel->Top = 80;
AlertPanel->Left = 1;
AlertPanel->Top = 80;
FindPanel->Left = 1;
FindPanel->Top = 80;
LogPanel->Left = 1;
LogPanel->Top = 80;
ComboBox1->ItemIndex = 0;
Panel3->Caption="          База данных:      " + ComboBox1->Text + " ";
SpeedButton1->Click();
SpeedButton1->Down = true;

TDateTime dt;
dt = Now();
unsigned short year, month, day;
dt.DecodeDate(&year, &month, &day);
ComboBox6->ItemIndex = month - 1;
UpDown4->Position = year;
ComboBox5->ItemIndex = 0;
ComboBox6Change(ComboBox6);
ComboBox5Change(ComboBox5);
}
//-----
void __fastcall TForm1::SpeedButton2Click(TObject *Sender)
{
AlertPanel->Visible = false;
SettingPanel->Visible = false;
FindPanel->Visible = false;
LogPanel->Visible = false;
DbPanel->Visible= true;
}
//-----
void __fastcall TForm1::SpeedButton3Click(TObject *Sender)
{
AlertPanel->Visible = false;
DbPanel->Visible= false;
FindPanel->Visible = false;
LogPanel->Visible = false;
Edit3->Text = "";
SettingPanel->Visible = true;
}
//-----
void __fastcall TForm1::SpeedButton1Click(TObject *Sender)

```

```

{
    SettingPanel->Visible = false;
    DbPanel->Visible= false;
    FindPanel->Visible = false;
    LogPanel->Visible = false;
    AlertPanel->Visible = true;
}
//-----
void __fastcall TForm1::SpeedButton4Click(TObject *Sender)
{
    ExitWhile = false;
    if (SpeedButton4->Caption == "Установить соединение") {
        Form2->ShowModal();
        if (Form2->ModalResult == mrOk) {
            hCom = CreateFile((Form2->ListBox1->Items->Strings[Form2->
                >ListBox1->ItemIndex]).c_str(),
                GENERIC_READ, 0, NULL,
                OPEN_EXISTING, FILE_FLAG_OVERLAPPED, NULL);
            if (hCom == INVALID_HANDLE_VALUE) return;
            GetCommState(hCom, &dd);
            dd.BaudRate = 9600;
            SetCommState(hCom, &dd);
            o.hEvent = CreateEvent(NULL, FALSE, FALSE, NULL);
            ReadFile(hCom, buf, 13, &res, &o);
            Form4->ProgressBar1->Position = 0;
            Form1->Enabled = false;
            Form4->Show();
            Application->ProcessMessages();
            for (int i=1; i <= 650; i++) {
                Sleep(100);
                if (GetOverlappedResult(hCom, &o, &res, 0)) {
                    Form1->Enabled = true;
                    Form4->Hide();
                    Query1->SQL->Text = "INSERT INTO Alert (Object, DateTime,
Obrabotana) VALUES (\\"Установлено соединение с системой Дунай\\" , \\"
+ Now().DateTimeString() + \\", \\"y\\");
                    Query1->ExecSQL();
                    break;
                };
                Form4->ProgressBar1->Position = (i*1000) / 650;
                Application->ProcessMessages();
                if (i == 650) {
                    FileClose((int)hCom);
                    ExitWhile = true;
                    Form1->Enabled = true;
                    Form4->Hide();
                    Query1->SQL->Text = "INSERT INTO Alert (Object, DateTime,
Obrabotana) VALUES (\\"Попытка установить соединение с системой Дунай
провалилась\\" , \\" + Now().DateTimeString() + \\", \\"y\\");
                    Query1->ExecSQL();
                    return ;
                }
            }
            ReadFile(hCom, buf, 13, &res, &o);
            Timer1->Enabled = true;
            SpeedButton4->Caption = "Разорвать соединение";
            N5->Caption = SpeedButton4->Caption;
            if (!RightDisconnect) {

```

```

        SpeedButton4->Enabled = false;
        N5->Enabled = false;
    }
    while(1) {
        if (ExitWhile) break;
        if (ExitProgram) {
            FileClose((int)hCom);
            return;
        }
        Application->ProcessMessages();
        if (GetOverlappedResult(hCom, &o, &res, 0)) {
/*
            if (CheckBox7->Checked) {
                CheckBox7->Checked = false;
                buf[0]='1'; buf[1]='1'; buf[2]='0'; buf[3]='0';
                buf[4]='1'; buf[5]='0'; buf[6]='0'; buf[7]='1';
                buf[8]='0'; buf[9]='5'; buf[10]='1'; buf[11]='8';
                buf[12]=13;*/
                if (AnsiString(buf) != "FF000000001E\0xD") {
                    AnsiString s, s1, s2, s3, s4, s5;
                    s1 = AnsiString("0x")+buf[0];
                    s2 = AnsiString("0x")+buf[1];
                    s3 = AnsiString("0x")+buf[2]+buf[3]+buf[4]+buf[5];
                    s4 = AnsiString("0x")+buf[6]+buf[7];
                    s5 = AnsiString("0x")+buf[8]+buf[9];
                    s = IntToStr(StrToInt(s1))+ "." +
                        IntToStr(StrToInt(s2))+ "." +
                        IntToStr(StrToInt(s3))+ "." +
                        IntToStr(StrToInt(s4))+ "." +
                        IntToStr(StrToInt(s5));
                    tZones->Close();
                    tZones->Filter = "Code='"+s+"'";
                    tZones->Filtered = true;
                    tZones->Open();
                    if (tZones->RecordCount != 0) {
                        AnsiString s = "INSERT INTO Alert (Object, Location,
Zone,      DatTime,      Obrabotana)      VALUES      (\\" + tZones-
>FieldByName("lokObject")->AsString +
                        "\",      \\" + tZones-
>FieldByName("Lacation")->AsString +
                        "\",      \\" + tZones-
>FieldByName("Code")->AsString +
                        "\",      \\" + Now().DateTimeString() +
                        "\",      \\"n\");

                        Query1->SQL->Text = s;
                        Query1->ExecSQL();
                        qAlerts->DisableControls();
                        qAlerts->Close();
                        qAlerts->Open();
                        qAlerts->EnableControls();
                        TrevogaON();
                        SpeedButton1->Click();
                        SpeedButton1->Down = true;
                    }
                    tZones->Close();
                    tZones->Filter = "";
                    tZones->Filtered = false;
                    tZones->Open();
                }
            }
        }
    }
}

```

```

//          if (!WriteFile(hCom, &buf2, 1, &res, 0)) Message-
Box(0,"error","error",0);;
        WriteFile(hCom, &buf2, 1, &res, &o);
        while (!GetOverlappedResult(hCom, &o, &res, 0)) ;
        ReadFile(hCom, buf, 13, &res, &o);
        Timer1->Enabled = false;
        Timer1->Enabled = true;
        if (Panel6->Font->Color == clWhite) {
            Query1->SQL->Text = "INSERT INTO Alert (Object, DateTime,
Obrabotana) VALUES (\\"Связь с системой Дунай восстановилась\\" , \\" +
Now().DateTimeString() + "\\", \\"y\\");
            Query1->ExecSQL();
        }
        NoconnectOFF();
    }
}
}
else {
    if (!RightDisconnect) return;
    FileClose((int)hCom);
    NoconnectOFF();
    SpeedButton4->Caption = "Установить соединение";
    N5->Caption = SpeedButton4->Caption;
    Query1->SQL->Text = "INSERT INTO Alert (Object, DateTime, Obrabota-
na) VALUES (\\"Пользователь разорвал связь с системой Дунай\\" , \\" +
Now().DateTimeString() + "\\", \\"y\\");
    Query1->ExecSQL();
    Timer1->Enabled = false;
    ExitWhile = true;
}
}
//-----
void __fastcall TForm1::BitBtn3Click(TObject *Sender)
{
    if (qAlerts->RecordCount != 0) {
        if (Application->MessageBox("Тревога обработана ?", "Подтвердите",
1) != IDOK)
            return;
        Query1->SQL->Text = "UPDATE Alert SET Obrabotana = \\"y\\" WHERE Kod
="+qAlerts->FieldName("Kod")->AsString;
        Query1->ExecSQL();
        qAlerts->DisableControls();
        qAlerts->Close();
        qAlerts->Open();
        qAlerts->EnableControls();
    }
    if (qAlerts->RecordCount == 0)
        TrevogaOFF();
}
//-----
void __fastcall TForm1::SpeedButton5Click(TObject *Sender)
{
    Query1->SQL->Text = "INSERT INTO Alert (Object, DateTime, Obrabotana)
VALUES (\\"Выход из программы\\" , \\" + Now().DateTimeString() + "\\",
\\"y\\");
    Query1->ExecSQL();
}

```

```

    ExitProgram = true;
    Close();
}
//-----
void __fastcall TForm1::N3Click(TObject *Sender)
{
    SpeedButton2->Click();
    SpeedButton2->Down = true;
}
//-----
void __fastcall TForm1::N2Click(TObject *Sender)
{
    SpeedButton1->Click();
    SpeedButton1->Down = true;
}
//-----
void __fastcall TForm1::N4Click(TObject *Sender)
{
    SpeedButton3->Click();
    SpeedButton3->Down = true;
}
//-----
void __fastcall TForm1::N6Click(TObject *Sender)
{
    SpeedButton5Click(SpeedButton5);
}
//-----
void __fastcall TForm1::N5Click(TObject *Sender)
{
    SpeedButton4Click(SpeedButton4);
}
//-----
void __fastcall TForm1::Timer1Timer(TObject *Sender)
{
    if (Panel6->Font->Color == clWhite) return;
    Query1->SQL->Text = "INSERT INTO Alert (Object, DatTime, Obrabotana)
VALUES      (\\"Потеряна связь с системой Дунай\\"", \"" +
Now().DateTimeString() + "\", \\"y\\")";
    Query1->ExecSQL();
    NoconnectON();
}
//-----
void __fastcall TForm1::BitBtn1Click(TObject *Sender)
{
    DataSource1->DataSet->Insert();
    BitBtn14Click(BitBtn14);
}
//-----
void __fastcall TForm1::BitBtn2Click(TObject *Sender)
{
    DataSource1->DataSet->Delete();
}
//-----
void __fastcall TForm1::tZonesAfterScroll(TDataSet *DataSet)
{
    Memo1->Lines->Clear();
    Memo1->Lines->Text      =      tZones->FieldName("Lacation")-
>AsString;

```

```

}
//-----
void __fastcall TForm1::tZonesBeforeScroll(TDataSet *DataSet)
{
    if (Memol->Lines->Text == "") return;
    tZones->Edit();
    tZones->FieldByName("Lacation")->AsString = Memol->Lines-
>Text;
    tZones->Post();
}
//-----
void __fastcall TForm1::vcdc(TObject *Sender)
{
    SettingPanel->Visible = false;
    DbPanel->Visible = false;
    AlertPanel->Visible = false;
    LogPanel->Visible = false;
    FindPanel->Visible = true;
}
//-----
void __fastcall TForm1::BitBtn7Click(TObject *Sender)
{
    Finded1->Clear();
    Finded2->Clear();
    StringGrid1->RowCount = 2;
    StringGrid1->Cells[0][0] = "Найденная информация";
    StringGrid1->Cells[1][0] = "Объект";
    StringGrid1->Cells[2][0] = "Поле";
    StringGrid1->Cells[0][1] = "";
    StringGrid1->Cells[1][1] = "";
    StringGrid1->Cells[2][1] = "";
}
//-----
void TForm1::FindFields(TDataSet* dataset, AnsiString datasetName, An-
siString field, bool like, AnsiString Key)
{
    AnsiString sField;
    Key = Key.UpperCase();
    AnsiString s = dataset->FieldByName(field)->DisplayLabel;
    dataset->First();
    while (!dataset->Eof) {
        Application->ProcessMessages();
        if (StopFind) return;
        sField = dataset->FieldByName(field)->AsString;
        sField = sField.UpperCase();
        if (!like && sField == Key || like && sField.Pos(Key) != 0) {
            StringGrid1->Cells[0][StringGrid1->RowCount-1] = dataset-
>FieldByName(field)->AsString;
            StringGrid1->Cells[1][StringGrid1->RowCount-1] = datasetName;
            StringGrid1->Cells[2][StringGrid1->RowCount-1] = s;
            int k = dataset->Fields->IndexOf(dataset->FieldByName(field));
            Finded1->Add(AnsiString(dataset->RecNo - 1));
            Finded2->Add(IntToStr(k));
            StringGrid1->RowCount++;
            StringGrid1->Cells[0][StringGrid1->RowCount-1] = "";
            StringGrid1->Cells[1][StringGrid1->RowCount-1] = "";
            StringGrid1->Cells[2][StringGrid1->RowCount-1] = "";
        }
    }
}

```

```

        dataset->Next();
    }
    return;
}
void __fastcall TForm1::BitBtn5Click(TObject *Sender)
{
    StopFind = false;
    BitBtn5->Enabled = false;
    BitBtn7Click(BitBtn7);
    FindFields(tObject, "Объекты", "Full_Name", 1, Edit2->Text);
    FindFields(tObject, "Объекты", "Owner", 1, Edit2->Text);
    FindFields(tObject, "Объекты", "Chif_Name", 1, Edit2->Text);
    FindFields(tObject, "Объекты", "Chif_PhoneW", 0, Edit2->Text);
    FindFields(tObject, "Объекты", "Chif_PhoneH", 0, Edit2->Text);
    FindFields(tObject, "Объекты", "Chif_PhoneM", 0, Edit2->Text);
    FindFields(tObject, "Объекты", "Depu_Name", 1, Edit2->Text);
    FindFields(tObject, "Объекты", "Depu_PhoneW", 0, Edit2->Text);
    FindFields(tObject, "Объекты", "Depu_PhoneH", 0, Edit2->Text);
    FindFields(tObject, "Объекты", "Depu_PhoneM", 0, Edit2->Text);
    FindFields(tObject, "Объекты", "Rentor", 1, Edit2->Text);
    FindFields(tObject, "Объекты", "RChif_Name", 1, Edit2->Text);
    FindFields(tObject, "Объекты", "RChif_PhoneW", 0, Edit2->Text);
    FindFields(tObject, "Объекты", "RChif_PhoneH", 0, Edit2->Text);
    FindFields(tObject, "Объекты", "RChif_PhoneM", 0, Edit2->Text);
    FindFields(tObject, "Объекты", "RDepu_Name", 1, Edit2->Text);
    FindFields(tObject, "Объекты", "RDChif_PhoneW", 0, Edit2->Text);
    FindFields(tObject, "Объекты", "RDChif_PhoneH", 0, Edit2->Text);
    FindFields(tObject, "Объекты", "RDChif_PhoneM", 0, Edit2->Text);
    FindFields(tBranche, "Тип объекта", "Name", 1, Edit2->Text);
    FindFields(tStreet, "Улицы", "Name", 1, Edit2->Text);
    FindFields(tEquipment, "Оборудование", "Name", 1, Edit2->Text);
    FindFields(tServices, "Обслуживающая организация", "Name", 1, Edit2->Text);
    FindFields(tServices, "Обслуживающая организация", "Chif_Name", 1, Edit2->Text);
    FindFields(tServices, "Обслуживающая организация", "Chif_PhoneW", 0, Edit2->Text);
    FindFields(tServices, "Обслуживающая организация", "Chif_PhoneH", 0, Edit2->Text);
    FindFields(tServices, "Обслуживающая организация", "Chif_PhoneM", 0, Edit2->Text);
    BitBtn5->Enabled = true;
}
//-----
void __fastcall TForm1::BitBtn6Click(TObject *Sender)
{
    StopFind = true;
}
//-----
void __fastcall TForm1::BitBtn8Click(TObject *Sender)
{
    __try {
        TTable* t1 = NULL;
        AnsiString s1 = StringGrid1->Cells[1][StringGrid1->Row];
        if (s1 == "Объекты")
            t1 = tObject;
        if (s1 == "Улицы")
            t1 = tStreet;
    }
}

```

```

if (s1 == "Оборудование")
    t1 = tEquipment;
if (s1 == "Тип объекта")
    t1 = tBranche;
if (s1 == "Обслуживающая организация")
    t1 = tServices;
if (s1 == "Шлейфы")
    t1 = tZones;
if (t1 == NULL) return;
SpeedButton2->Click();
SpeedButton2->Down = true;
ComboBox1->ItemIndex = ComboBox1->Items->IndexOf(s1);
ComboBox1Change(ComboBox1);
    t1->First();
    t1->MoveBy(StrToInt(Finded1->Strings[StringGrid1->Row-1]));
    int k = StrToInt(Finded2->Strings[StringGrid1->Row-1]);
    DBGrid1->SelectedIndex = k;
    DBGrid1->SetFocus();
}
__finally {};
}
//-----
void __fastcall TForm1::SpeedButton7Click(TObject *Sender)
{
    qLog->Close();
    qLog->Open();
    SettingPanel->Visible = false;
    DbPanel->Visible = false;
    AlertPanel->Visible = false;
    FindPanel->Visible = false;
    LogPanel->Visible = true;
}
//-----
void __fastcall TForm1::BitBtn9Click(TObject *Sender)
{
    // Query1->SQL->Text = "DELETE FROM Alert WHERE DateTime <= \"\" +
    DateTimePicker1->Date.DateString() + "\"";
    /* Query1->ExecSQL();
    qLog->DisableControls();
    qLog->Close();
    qLog->Open();
    qLog->EnableControls(); */
    int i = (int)ShellExecute(0,"open", "Backup.exe", "" , "Plu-
gins", SW_SHOW);
    // if (i==ERROR_FILE_NOT_FOUND)
    //     MessageBox(0,"dfdfd","dfdfd",0);
}
//-----
void __fastcall TForm1::BitBtn4Click(TObject *Sender)
{
    AnsiString s;
    if(qAlerts->RecordCount == 0) return;
    Alert->Memo1->Lines->Clear();
    tObject->Close();
    tObject->Filter = "Full_Name='" + qAlerts-
>FieldByName("Object")->AsString + "'";
    tObject->Filtered = true;
    tObject->Open();

```

```

        s = "Тревога на объекте: " + tObject->FieldByName("Full_Name")->AsString;
Alert->Memol->Lines->Add(s);
        s = "Отрасль: " + tObject->FieldByName("BranchLook")->AsString;
Alert->Memol->Lines->Add(s);
Alert->Memol->Lines->Add("");
        s = "Улица: " + tObject->FieldByName("StreetLook")->AsString;
Alert->Memol->Lines->Add(s);
        s = "Дом: " + tObject->FieldByName("A_Home")->AsString +
            ". Этаж: " + tObject->FieldByName("A_Flow")->AsString +
            ". Помещение: " + tObject->FieldByName("A_Apartment")->AsString +
            ". Подъезд: " + tObject->FieldByName("A_Doorway")->AsString + ".";
Alert->Memol->Lines->Add(s);
Alert->Memol->Lines->Add("");
        s = "Тел. дежурный: " + tObject->FieldByName("Duty_Phone")->AsString;
Alert->Memol->Lines->Add(s);
Alert->Memol->Lines->Add("");
Add("_____");
        s = "Собственник: " + tObject->FieldByName("Owner")->AsString;
Alert->Memol->Lines->Add(s);
        s = "Руководитель: " + tObject->FieldByName("Chif_Name")->AsString;
Alert->Memol->Lines->Add(s);
        s = "Тел. рабочий: " + tObject->FieldByName("Chif_PhoneW")->AsString;
Alert->Memol->Lines->Add(s);
        s = "Тел. домашний: " + tObject->FieldByName("Chif_PhoneH")->AsString;
Alert->Memol->Lines->Add(s);
        s = "Тел. мобильный: " + tObject->FieldByName("Chif_PhoneM")->AsString;
Alert->Memol->Lines->Add(s);
        s = "Заместитель: " + tObject->FieldByName("Depu_Name")->AsString;
Alert->Memol->Lines->Add(s);
        s = "Тел. рабочий: " + tObject->FieldByName("Depu_PhoneW")->AsString;
Alert->Memol->Lines->Add(s);
        s = "Тел. домашний: " + tObject->FieldByName("Depu_PhoneH")->AsString;
Alert->Memol->Lines->Add(s);
        s = "Тел. мобильный: " + tObject->FieldByName("Depu_PhoneM")->AsString;
Alert->Memol->Lines->Add(s);
Alert->Memol->Lines->Add("");
Add("_____");
        s = "Арендодатель: " + tObject->FieldByName("Rentor")->AsString;
Alert->Memol->Lines->Add(s);
        s = "Руководитель: " + tObject->FieldByName("RChif_Name")->AsString;
Alert->Memol->Lines->Add(s);
        s = "Тел. рабочий: " + tObject->FieldByName("RChif_PhoneW")->AsString;
Alert->Memol->Lines->Add(s);

```

```

        Alert->Memo1->Lines->Add(s);
        s = "Тел. домашний: " + tObject->FieldByName("RChif_PhoneH") -
>AsString;
        Alert->Memo1->Lines->Add(s);
        s = "Тел. мобильный: " + tObject->FieldByName("RChif_PhoneM") -
>AsString;
        Alert->Memo1->Lines->Add(s);
        s = "Заместитель: " + tObject->FieldByName("RDepu_Name") -
>AsString;
        Alert->Memo1->Lines->Add(s);
        s = "Тел. рабочий: " + tObject->FieldByName("RDChif_PhoneW") -
>AsString;
        Alert->Memo1->Lines->Add(s);
        s = "Тел. домашний: " + tObject->FieldByName("RDChif_PhoneH") -
>AsString;
        Alert->Memo1->Lines->Add(s);
        s = "Тел. мобильный: " + tObject->FieldByName("RDChif_PhoneM") -
>AsString;
        Alert->Memo1->Lines->Add(s);
        s = "Тел. дежурный: " + tObject->FieldByName("RDuty_Phone") -
>AsString;
        Alert->Memo1->Lines->Add(s);
        Alert->Memo1->Lines-
>Add("_____");
        s = "Обслуживающая организация: " + tObject-
>FieldByName("ServicesLook")->AsString;
        Alert->Memo1->Lines->Add(s);
        s = "Оборудование: " + tObject->FieldByName("EquipmentLook") -
>AsString;
        Alert->Memo1->Lines->Add(s);
        Alert->Show();
        tObject->Close();
        tObject->Filter = "";
        tObject->Filtered = false;
        tObject->Open();
    }
    //-----
    void __fastcall TForm1::N7Click(TObject *Sender)
    {
        SpeedButton6->Click();
        SpeedButton6->Down = true;
    }
    //-----
    void __fastcall TForm1::N8Click(TObject *Sender)
    {
        SpeedButton7->Click();
        SpeedButton7->Down = true;
    }
    //-----
    void __fastcall TForm1::tAccessAfterScroll(TDataSet *DataSet)
    {
        if (tAccess->FieldByName("Right1")->AsString == "write")
            ComboBox2->ItemIndex = 0;
        if (tAccess->FieldByName("Right1")->AsString == "read")
            ComboBox2->ItemIndex = 1;
        if (tAccess->FieldByName("Right1")->AsString == "")
            ComboBox2->ItemIndex = 2;
        if (tAccess->FieldByName("Right2")->AsString == "write")

```

```

        ComboBox3->ItemIndex = 0;
        if (tAccess->FieldByName("Right2")->AsString == "read")
            ComboBox3->ItemIndex = 1;
        if (tAccess->FieldByName("Right2")->AsString == "")
            ComboBox3->ItemIndex = 2;
        if (tAccess->FieldByName("Right3")->AsString == "yes")
            ComboBox4->ItemIndex = 0;
        if (tAccess->FieldByName("Right3")->AsString == "")
            ComboBox4->ItemIndex = 1;
        Edit1->Text = tAccess->FieldByName("Password")->AsString;
        if (tAccess->FieldByName("Root")->AsBoolean) Label7->Font-
>Color = clMaroon; else Label7->Font->Color = clGray;
    }
    //-----
    void __fastcall TForm1::ComboBox2Change(TObject *Sender)
    {
        AnsiString f;
        tAccess->Edit();
        if (Sender == ComboBox2) f = "Right1"; else f = "Right2";
        if ((TComboBox*)(Sender)->ItemIndex == 0)
            tAccess->FieldByName(f)->AsString = "write";
        if ((TComboBox*)(Sender)->ItemIndex == 1)
            tAccess->FieldByName(f)->AsString = "read";
        if ((TComboBox*)(Sender)->ItemIndex == 2)
            tAccess->FieldByName(f)->AsString = "";
        tAccess->Post();
    }
    //-----
    void __fastcall TForm1::BitBtn11Click(TObject *Sender)
    {
        if (tAccess->FieldByName("Root")->AsBoolean) return;
        tAccess->Delete();
        DBLookupListBox1->KeyValue = tAccess->FieldByName("UserName")-
>AsString;
    }
    //-----
    void __fastcall TForm1::BitBtn10Click(TObject *Sender)
    {
        if (Edit3->Text == "") return;
        tAccess->SetKey();
        tAccess->FieldByName("UserName")->AsString = CurrentUser;
        tAccess->GotoKey();
        AnsiString s = tAccess->FieldByName("Setting")->AsString;
        tAccess->Insert();
        tAccess->FieldByName("UserName")->AsString = Edit3->Text;
        tAccess->FieldByName("Password")->AsString = "";
        tAccess->FieldByName("Root")->AsBoolean = false;
        tAccess->FieldByName("Setting")->AsString = s;
        tAccess->Post();
        DBLookupListBox1->KeyValue = tAccess->FieldByName("UserName")-
>AsString;
        Edit3->Text = "";
    }
    //-----

    void __fastcall TForm1::BitBtn12Click(TObject *Sender)
    {
        AnsiString s = Edit1->Text;

```

```

        if (s == "") return;
        int c = tAccess->RecNo;
        tAccess->First();
        for (int i=0; i < tAccess->RecordCount; i++) {
            if (tAccess->FieldByName("Password")->AsString.UpperCase() ==
s.UpperCase()) {
                tAccess->First();
                tAccess->MoveBy(c-1);
                return;
            }
            tAccess->Next();
        }
        tAccess->First();
        tAccess->MoveBy(c-1);
        tAccess->Edit();
        tAccess->FieldByName("Password")->AsString = s;
        tAccess->Post();
        tAccess->First();
        tAccess->MoveBy(c-1);
    }
    //-----
void __fastcall TForm1::BitBtn13Click(TObject *Sender)
{
    SaveSetting();
}
    //-----
void __fastcall TForm1::BitBtn14Click(TObject *Sender)
{
    DBGrid1->ReadOnly = false;
    DataSource1->DataSet->Edit();
    DBGrid1->SetFocus();
    BitBtn1->Enabled = false;
    BitBtn2->Enabled = false;
    BitBtn14->Enabled = false;
}
    //-----
void __fastcall TForm1::tObjectAfterPost(TDataSet *DataSet)
{
    DBGrid1->ReadOnly = true;
    BitBtn1->Enabled = true;
    BitBtn2->Enabled = true;
    BitBtn14->Enabled = true;
}
    //-----
void __fastcall TForm1::DBGrid1KeyPress(TObject *Sender, char
&Key)
{
    if (!BitBtn14->Enabled) return ;
    dynamic_cast<TTable*>(DataSource1->DataSet)->SetKey();
    if (DataSource1->DataSet->Name == "tObject")
        DataSource1->DataSet->FieldByName("Full_Name")-
>AsString = Key;
    if (DataSource1->DataSet->Name == "tStreet")
        DataSource1->DataSet->FieldByName("Name")-
>AsString = Key;
    if (DataSource1->DataSet->Name == "tBranche")
        DataSource1->DataSet->FieldByName("Name")-
>AsString = Key;

```

```

        if (DataSource1->DataSet->Name == "tEquipment")
            DataSource1->DataSet->FieldByName("Name") -
>AsString = Key;
        if (DataSource1->DataSet->Name == "tServices")
            DataSource1->DataSet->FieldByName("Name") -
>AsString = Key;
        ((TTable*)(DataSource1->DataSet))->GotoNearest();
    }
    //-----
void __fastcall TForm1::tObjectAfterCancel(TDataSet *DataSet)
{
    DBGrid1->ReadOnly = true;
    BitBtn1->Enabled = true;
    BitBtn2->Enabled = true;
    BitBtn14->Enabled = true;
}
    //-----
void __fastcall TForm1::ComboBox4Change(TObject *Sender)
{
    tAccess->Edit();
    if ((TComboBox*)(Sender))->ItemIndex == 0)
        tAccess->FieldByName("Right3")->AsString = "yes";
    if ((TComboBox*)(Sender))->ItemIndex == 1)
        tAccess->FieldByName("Right3")->AsString = "";
    tAccess->Post();
}
    //-----
void __fastcall TForm1::Edit4Exit(TObject *Sender)
{
    Edit4->PasswordChar = '*';
}
    //-----
void __fastcall TForm1::Edit4Enter(TObject *Sender)
{
    Edit4->PasswordChar = 0;
}
    //-----
void __fastcall TForm1::BitBtn15Click(TObject *Sender)
{
    if (ComboBox1->Text == "Объекты")
        DataSource1->DataSet = tObject;
    TTable* t1;
    PrintForm->RichEdit1->Clear();
    t1 = (TTable*)DataSource1->DataSet;
    if (ComboBox1->Text == "Улицы" || ComboBox1->Text == "Оборудо-
вание" || ComboBox1->Text == "Тип объекта") {
        PrintForm->RichEdit1->Lines->Add(ComboBox1->Text);
        PrintForm->RichEdit1->Lines->Add("");
        PrintForm->RichEdit1->Lines->Add("");
        t1->First();
        for (int i=0; i<t1->RecordCount; i++) {
            PrintForm->RichEdit1->Lines->Add(t1->FieldByName("Name") -
>AsString);
            t1->Next();
        }
    }
    if (ComboBox1->Text == "Обслуживающая организация") {
        PrintForm->RichEdit1->Lines->Add(ComboBox1->Text);
    }
}

```

```

        PrintForm->RichEdit1->Lines->Add("");
        PrintForm->RichEdit1->Lines->Add("");
        t1->First();
        for (int i=0; i<t1->RecordCount; i++) {
            PrintForm->RichEdit1->Lines->Add("");
            PrintForm->RichEdit1->Lines-
>Add("_____");
            PrintForm->RichEdit1->Lines->Add("");
            PrintForm->RichEdit1->Lines->Add(t1->FieldByName("Name") -
>AsString);
            PrintForm->RichEdit1->Lines->Add("Руководитель " + t1-
>FieldByName("Chif_Name")->AsString);
            PrintForm->RichEdit1->Lines->Add("Рабочий телефон " + t1-
>FieldByName("Chif_PhoneW")->AsString);
            PrintForm->RichEdit1->Lines->Add("Домашний телефон " + t1-
>FieldByName("Chif_PhoneH")->AsString);
            PrintForm->RichEdit1->Lines->Add("Мобильный телефон " + t1-
>FieldByName("Chif_PhoneM")->AsString);
            t1->Next();
        }
    }
    if (ComboBox1->Text == "Объекты") {
        ChoiseObject->Show();
        return;
    }
    /* if (ComboBox1->Text == "Шлейфы")
        DataSource1->DataSet = tZones;
    */
    PrintForm->Show();
}
//-----
void TForm1::TrevogaON()
{
    Panel7->Font->Color = clWhite;
    Panel7->Color = clRed;
    PlaySound("Alarm.wav", 0, SND_ASYNC | SND_LOOP);
}
void TForm1::TrevogaOFF()
{
    Panel7->Font->Color = clGray;
    Panel7->Color = clBtnFace;
    PlaySound(0, 0, 0);
}
void TForm1::NoconnectON()
{
    Panel6->Font->Color = clWhite;
    Panel6->Color = clRed;
    if (Panel7->Color != clRed)
        PlaySound("Alarm.wav", 0, SND_ASYNC);
}
void TForm1::NoconnectOFF()
{
    Panel6->Font->Color = clGray;
    Panel6->Color = clBtnFace;
}
void __fastcall TForm1::ComboBox5Change(TObject *Sender)
{
    AnsiString s;

```

```

qLog->Close();
qLog->SQL->Clear();
s = "SELECT * FROM Alert WHERE ";
qLog->SQL->Add(s);
qLog->SQL->Add(constr);
switch (ComboBox5->ItemIndex) {
    case 0: s = "(NOT Object IS NULL)"; break;
    case 1: s = "((NOT Object LIKE \"Пользователь%\") AND (NOT
Object LIKE \"Выход%\") AND (NOT Object LIKE \"Попытка%\") AND (NOT
Object LIKE \"Потеряна%\") AND (NOT Object like \"Связь с системой Ду-
най%\") AND (NOT Object LIKE \"Установлено%\"))"; break;
    case 2: s = "((Object like \"Связь с системой Дунай%\") OR
(Object LIKE \"Потеряна%\") OR (Object LIKE \"Установлено%\") OR (Ob-
ject LIKE \"Попытка%\") OR (Object LIKE \"Пользователь разорвал%\"))";
break;
    case 3: s = "((Object LIKE \"Пользователь%запустил%\") OR
(Object LIKE \"Выход%\"))"; break;
}
qLog->SQL->Add(s);
qLog->SQL->Add("ORDER BY DateTime");
qLog->Open();
}
//-----
void __fastcall TForm1::BitBtn16Click(TObject *Sender)
{
    PrintForm->RichEdit1->Clear();
    PrintForm->RichEdit1->Lines->Add("Журнал");
    PrintForm->RichEdit1->Lines->Add(ComboBox5->Text);
    PrintForm->RichEdit1->Lines->Add("");
    PrintForm->RichEdit1->Lines->Add("");
    qLog->First();
    for (int i=0; i<qLog->RecordCount; i++) {
        PrintForm->RichEdit1->Lines->Add(qLog->
>FieldByName("DateTime")->AsString + " " + qLog->
>FieldByName("Object")->AsString + " " + qLog->
>FieldByName("Location")->AsString + " " + qLog->
>FieldByName("Zone")->AsString);
        qLog->Next();
    }
    PrintForm->Show();
}
//-----
void __fastcall TForm1::ComboBox6Change(TObject *Sender)
{
    if (ComboBox6->ItemIndex + 1 != 12)
        constr = " (DateTime >= \"1.\" + AnsiString(ComboBox6->
ItemIndex + 1) + \".\" + AnsiString(UpDown4->Position) + " 00:00:00\")
AND (DateTime <= \"1.\" + AnsiString(ComboBox6->ItemIndex + 2) + \".\" +
AnsiString(UpDown4->Position) + " 00:00:00\") AND ";
    else
        constr = " (DateTime >= \"1.\" + AnsiString(ComboBox6->
ItemIndex + 1) + \".\" + AnsiString(UpDown4->Position) + " 00:00:00\")
AND (DateTime <= \"31.\" + AnsiString(ComboBox6->ItemIndex + 1) + \".\" +
AnsiString(UpDown4->Position) + " 23:59:59\") AND ";
    ComboBox5Change(ComboBox5);
}

```