Міністерство освіти і науки України
Український державний університет науки і технологій

Факультет «Комп'ютерних технологій і систем»
(назва факультету)

Кафедра «Електронні обчислювальні машини»
(повна назва кафедри)

Пояснювальна записка
до кваліфікаційної роботи
магістра
(ступінь вищої освіти)

на тему: Розробка АРМа оператора сортувальної гірки

за освітньою програмою Комп'ютерна інженерія
зі спеціальності: 123 Комп'ютерна інженерія
(шифр і назва спеціальності)

Виконав: студент групи: КС2326(8КС)
(підпис студента) /Роман КІНЗЯБУЛАТОВ /
(Ім'я ПРІЗВИЩЕ)

Керівник: (підпис) /доцент, Олег ЄГОРОВ /
(посада, Ім'я ПРІЗВИЩЕ)

Нормоконтролер: (підпис) /доцент, Олег ЄГОРОВ /
(посада, Ім'я ПРІЗВИЩЕ)

Засвідчую, що у цій роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент (підпис)

Дніпро – 2025 рік

Ministry of Education and Science of Ukraine
Ukrainian State University of Science and Technologies

Faculty of Computer Technologies and Systems
(faculty)

Department of Electronic Computing Machines
(department)

Explanatory Note
to Master's Thesis
Master's degree
(higher education degree)

on the topic: Development of automated workstation for a hump yard operator

according to educational curriculum Computer Engineering

in the Speciality: 123 Computer Engineering
(speciality and its code)

Done by the student of the group: KS2326(8KS) / Roman Kinziabulatov /
(name, surname)

Scientific Supervisor: /docent, Oleh Yehorov /
(position, name, surname)

Normative controller : / docent, Oleh Yehorov /
(position, name, surname)

Dnipro – 2025

Міністерство освіти і науки України
Український державний університет науки і технологій

Факультет: Комп'ютерних технологій і систем
Кафедра: Електронні обчислювальні машини
Рівень вищої освіти: Магістр
Освітня програма: Комп'ютерна інженерія
Спеціальність: 123 Комп'ютерна інженерія
(шифр та назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Ігор Жуковицький
(підпис) (Ім'я ПРІЗВИЩЕ)
Дата

З А В Д А Н Н Я
на кваліфікаційну роботу магістра
(ступінь вищої освіти)

студенту Кінзябулатову Роману Ураловичу
(Прізвище, Ім`я По батькові)

1. Тема роботи: Розробка АРМа оператора сортувальної гірки

Керівник роботи: Єгоров Олег Йосипович, кандидат технічних наук, доцент
(Прізвище, Ім'я, По батькові, науковий ступінь, вчене звання)

затверджені наказом від "16" вересня 2024 р. № 1119

2. Строк подання студентом роботи: 21.01.2025 р.
3. Вихідні дані до роботи: технологічний процес - робоче місце оператора сортувальної гірки

4. Зміст пояснювальної записки (перелік питань, які потрібно опрацювати):
4.1 Дослідження інформаційних систем на СС
4.2 Порівняльний аналіз програмного тв апаратного забезпечення
4.3 Опис ТО та ІП АРМу
4.4 Програмно-апаратна реалізація АРМу

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Дослідження ІС на СС	15.11.2024р.	10%
2	Порівняльний аналіз ПЗ та АЗ АРМу	25.11.2024р.	20%
3	Опис структури ТО та ІП АРМу	01.12.2024р.	15%
4	Реалізація ПЗ та АЗ АРМу	12.01.2025р.	45%
5	Створення документації	18.01.2025р.	10%
6	Подання кваліфікаційної роботи до кафедри	21.01.2025р.	100%
7	Захист кваліфікаційної роботи на засіданні Екзаменаційної комісії	24.01.2025р.	

Студент _____ Роман Кінзябулатов
(підпис) (Ім'я ПРІЗВИЩЕ)

Керівник роботи _____ Олег Єгоров
(підпис) (Ім'я ПРІЗВИЩЕ)

Міністерство освіти і науки України
Український державний університет науки і технологій

Відгук керівника

кваліфікаційної роботи **магістра**
(ступінь вищої освіти)

Студент групи КС2326(8КС) Кінзябулатов Роман Уралович
(шифр групи) (Прізвище, Ім'я, По батькові)

Тема випускної роботи: Розробка АРМа оператора сортувальної гірки

1. Якісні відмінності кваліфікаційної роботи: середній рівень технічної реалізації та наукового підходу; глибоке розуміння технологічного процесу оператора та здатність застосовувати інструменти для створення програмних додатків; детальний аналіз технологічних процесів та інформаційних потоків.

2. Зауваження: в деяких розділах варто було б надати більш детальне пояснення щодо використаних алгоритмів, та інструментів.

3. Висновок щодо дотримання академічної доброчесності: всі принципи академічної доброчесності дотримані. Використані джерела були цитовані, а власні результати представлені чітко і обґрунтовано.

Комплексна оцінка кваліфікаційної роботи: робота відповідає вимогам до кваліфікаційних робіт, має високу практичну цінність. Робота виконана на середньому професійному рівні та заслуговує оцінки гарно.

Керівник: доцент Олег ЄГОРОВ
(посада) (Ім'я, ПРІЗВИЩЕ) (підпис)

Дата: 21.01.2025

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи магістра:

94с., 28 рис., 5 табл., 1 додаток, 14 джерел.

Об’єкт розробки – автоматизоване робоче місце чергового оператора сортувальної гірки. Програмно-апаратний продукт, розроблений під операційну систему Windows .

Мета роботи – автоматизація ряду технологічних завдань, що виконуються черговим оператором сортувальної гірки для розпуску составів.

Методи дослідження - аналіз роботи інформаційних систем верхнього рівня, що застосовуються на сортувальних станціях, та їх взаємозв'язок із підсистемами управління розпуску составів на сортувальній гірці. Вивчення інформаційно-керуючих потоків даних, що використовуються на сортувальній гірці для розпуску составів.

Результати роботи можуть стати основою для подальшої модернизації та автоматизації систем керування на сортувальних гірках, що входять до складу сортувальних станцій Укрзалізниці.

Ключові слова: АРМ, СОРТУВАЛЬНА ГІРКА, АВТОМАТИЗАЦІЯ, МОДЕРНИЗАЦІЯ, ІНФОРМАЦІЙНА СИСТЕМА, ПРОГРАМА РОЗПУСКУ

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

АРМ – Автоматизоване робоче місце

ІС – інформаційна система

СС – сортувальна станція

ПЗ – програмне забезпечення

АЗ – апаратне забезпечення

ТО – технологічна операція

ІП – інформаційний поток

ПП – Парк прибуття

СП – сортувальний парк

АСК – Автоматизована система керування

БД – База даних

ДСПГ – Диспетчерський пункт

ІТ – Інформаційні технології

ТП – Технологічний процес

СКБД – Система керування базою даних

АРМ ДСПГ – Автоматичне робоче місце чергового по гірці

ГАЦ - Гіркова автоматична централізація

АСК ВП УЗ - автоматизована систему вантажних перевезень Укрзалізниці

АСК ВП УЗ-Є – єдина автоматизована систему вантажних перевезень Укрзалізниці

ВСТУП

Програмне забезпеченяя (додаток) Арм оператора сортувальної гірки призначене для автоматизованого (ручного) управління процесом розформування поїздів на сортувальних гірках з метою автоматизації ряду технологічних завдань, що виконуються черговим оператором сортувальної гірки та покращення умов та якості роботи оперативного персоналу, підвищення продуктивності сортувальної гірки та станції в цілому.

Основні переваги впровадження даної системи:

- виключення ручних операцій при прийомі програми розпуску з АСК ВП УЗ і введення її в ГАЦ;
- скасування голосових команд ДСПГ для укладачів на вершині сортувальної гірки;
- зменшення експлуатаційних витрат за рахунок скорочення браку в роботі ДСПГ і операторської ланки, так як при пікових навантаженнях вони будуть керувати тільки швидкістю скочування відчепів;
- підвищення якості роботи укладачів на вершині сортувальної гірки, поліпшення умов їх праці;
- автоматичне ведення протоколу розпуску.

Опис роботи системи.

На основі даних натурного листа поїзду який прибув в парк прийому в АСК ВП УЗ формується сортувальний лист, який по каналу зв'язку передається в АРМ чергового по горці. Використовуючи сортувальний лист, в АРМі формується масив - ПРОГРАМА РОЗПУСКУ для даного поїзду, який містить порядковий номер відчепу, його довжину і маршрут слідування на підгіркову колію. При насуванні складу і підході його до вершини гірки по команді оператора скоригована програма розпуску пересилається з АРМа в мікропроцесорний контролер системи, який видає на колійні індикатори інформацію про довжинах перших двох відчепів для укладачів і на повторювач колійного індикатора на пульті оператора.

Одночасно з цим маршрути зазначених відчепів контролер пересилає в маршрутний накопичувач ГАЦ, і, як результат, - встановлюється маршрут для першого відчепу.

1 ДОСЛІДЖЕННЯ ІНФОРМАЦІЙНИХ СИСТЕМ, ЩО ЗАСТОСОВУЮТЬСЯ НА СОРТУВАЛЬНИХ СТАНЦІЯХ

1.1 Інформаційні системи верхнього рівня (АСК ВП УЗ, АСК ВП УЗ-Є)

Починаючи з 2007 року, залізничні вантажні перевезення України по-справжньому стали незалежними – було запроваджено нову автоматизовану систему вантажних перевезень Укрзалізниці (АСК ВП УЗ), розроблену фахівцями обчислювальних центрів залізниць України. Це призвело до забезпечення справжньої національної безпеки залізничного транспорту в галузі вантажних перевезень і є істотним кроком на шляху розвитку автоматизованих систем управління перевізним процесом залізничного транспорту. Зі створенням АСК ВП УЗ Укрзалізниця отримала нову сучасну систему управління вантажними перевезеннями, повністю незалежну від інших держав СНД та Балтії, яка за своїми справжніми можливостями та перспективами розвитку перекрила наявні закордонні аналоги.

Розробка та впровадження нової системи управління вантажними перевезеннями з повною підставою дозволяє стверджувати про перехід до глобальної інформатизації перевізного процесу. Універсальні принципи та методи, розроблені та реалізовані в АСК ВП УЗ, відкривають перспективи для перетворення автоматизованих систем на інтелектуальні системи підтримки прийняття рішень персоналом на різних рівнях.

Створена система дає можливість у повному обсязі оперативно зібрати та інтегрувати інформацію про всі необхідні технологічні та виробничі операції вантажних перевезень. На основі цих даних стає можливим застосувати нові інформаційні технології для фахівців головних управлінь, які безпосередньо беруть участь в організації, аналізі та управлінні експлуатаційною роботою, а також іншими сферами багатогранної діяльності підприємств Укрзалізниці.

Аавтоматизована система АСК ВП УЗ – це понад 150 різних технолого-економічних завдань моніторингу та керування вантажними перевезеннями, що відображають усі сторони процесів планування перевезень, обробку первинних перевізних документів, експлуатацію засобів перевезень, представлених різними

характеристиками вагонів, інших об'єктів. Кошти системи охоплюють весь життєвий цикл вагонів, локомотивів, нарахування тарифу за перевезення.

Організація перевізного процесу, його фактична реалізація, фінансово-економічні процеси, що супроводжують та забезпечують роботу залізничного транспорту при виконанні вантажних перевезень, стають ув'язаними в рамках єдиного середовища автоматизованих систем.

Зупинимося на деяких технологічних завданнях процесів перевезення, реалізованих серед системи АСК ВП УЗ.

Серед багатьох виділимо такі великі завдання:

- ведення попереджень на рух поїздів;

– місячне планування;

– оформлення перевізних документів та їх ув'язування з усіма етапами процесу перевезення;

- створення наскрізного логічного контролю процесів перевезення вантажів;

– облік роботи локомотивів та локомотивних бригад;

– використання картотек вагонів, контейнерів, під'їзних колій та інших.

Також створено необхідні умови для розробки та супроводу нових, сучасних автоматизованих систем управління залізничного транспорту України – підготовлено професійні кадри програмістів-розробників, було створено спеціалізоване державне підприємство з розробки АСК для залізничного транспорту України, ДП ПКТБ АСУЗТ, якому належало вирішувати численні комплексної автоматизації галузі.

У системі АСК ВП УЗ, побудованій на основі технології компонентно-орієнтованого програмування, передбачено можливості вдосконалення технологій процесу перевезення.

Введено компонент, який забезпечує вдосконалення технологій навантаження-вивантаження за рахунок створення та обробки відповідних подій.

Виконується всебічний облік та аналіз ефективності використання рухомого складу різних форм власності.

Важливим аспектом АСК ВП УЗ є і те, що на її основі забезпечується єдина мова для опису технологічних завдань виконання вантажних перевезень у термінах інформаційних повідомлень, що відображають ці процеси в моделях баз даних та баз знань АСК перевізного процесу.

Такий підхід відкриває нові шляхи вирішення завдань автоматизації на залізничному транспорті. В свій час АСК ВП УЗ на всіх залізницях обслуговувало понад 17,5 тис. користувачів України, щоденно обробляло понад 557 тис. технологічних документів, що створювало понад 600 тис. різноманітних довідок та звітів, що забезпечують ефективне функціонування залізничної галузі.

АСК ВП УЗ побудована з використанням найсучасніших світових досягнень у сфері інформаційних технологій, про що свідчать позитивні відгуки фахівців країн Західної Європи та комісій Міністерства транспорту та зв'язку України. Вже на початковому етапі впровадження АСК ВП УЗ дозволило не лише отримати економічний ефект, а й насамперед оптимізувати виробничі процеси, скоротити експлуатаційні витрати за рахунок уніфікації систем автоматизації, а також за рахунок незалежності замовлень на розробку та супровід систем управління з інших країн.

Інтеграція даних, різноманітних ресурсів та засобів, уніфікація завдань з управління на різних рівнях забезпечать значний технологічний, організаційний та економічний ефект, створюють можливість якісної та цілеспрямованої підготовки персоналу, забезпечать умови для підвищення конкурентної спроможності залізниць на ринку транспортних послуг.

Напрямок удосконалення автоматизованих систем залізничного транспорту пов'язується із завданням забезпечення перевезень на основі організації якісно нової взаємодії вантажовідправників та вантажоодержувачів із залізницями – виконання перевезень на базі «планування на добу». Сама можливість, а також ефективність такої роботи залізниць безпосередньо пов'язана з використанням автоматизованих систем керування. У цих умовах зростає роль методів та засобів підтримки прийняття оптимальних рішень щодо управління перевезеннями на

основі інформаційного та математичного моделювання всіх складових компонентів та умов процесів вантажних перевезень.

АСК ВП УЗ складається з комплексів, комплекси – із систем, системи – із завдань, завдання – із ресурсів. Завдання є найменшим компонентом, що самостійно впроваджується, ресурс – найменшим незалежно використовуваним компонентом.

Кожен компонент типу комплексу, системи чи завдання нумерується всередині вищестоящего компонента від 01 до 99. Приймається, що комплекси 01 – 19 є загальносистемними, тобто. е. що забезпечують загальне функціонування АСК ВП УЗ. Комплекси 20 – 99 зарезервовані для прикладних (функціональних) цілей, визначених документом «Основні напрямки розвитку інформатизації залізничного транспорту України».

В АСК ВП УЗ розглядаються 4 види ресурсів:

– документальні – проектні, технічні, технологічні, експлуатаційні та організаційні документи, що забезпечують розробку та функціонування системи;

– програмні – пакети, процедури, модулі та інші одиниці програмного забезпечення системи;

– інформаційні – моделі, схеми, таблиці та інші елементи БД, файли та файлові структури, інші утворення, призначені для постійного чи тимчасового зберігання інформації системи;

– технічні – сервери, ПК, мережеві пристрої, комплектуючі тощо.

Важливим для АСК ВП УЗ є поняття «тема». Тема - це функціональна освіта, що охоплює деяку прикладну область АСК ВП УЗ. Тема «накладається» на вищеописану схему декомпозиції, включає компоненти різних комплексів (зокрема – загальносистемних), які разом забезпечують досягнення цілей цієї теми. В АСК ВП УЗ першим етапом проектування будь-якої теми є визначення складу компонентів, необхідних її реалізації. Вибраний склад компонентів та необхідні від них функції фіксуються у спеціальному документі – аванпроекті на тему. Подальша технологія ведеться по-компонентно.

Вибір мов програмування в АСК ВП УЗ визначається рішенням про трирівневий спосіб роботи з інформацією: клієнт – рівень бізнес-логіки – сховище даних.

АСК ВП УЗ – це інтегроване середовище, що включає протоколи, інтерфейси, правила та відповідні програмні засоби для опису та розробки введення, зберігання та використання даних, а також створення додатків для забезпечення технологічних процесів залізничного транспорту. Це не лише конкретна автоматизована система, а певна технологія її безперервного розвитку – загальні правила, що регламентують способи нарощування її функціональних можливостей та орієнтовані на ці правила загальносистемні компоненти (не лише програмні, а й інформаційні, технічні, документальні).

Будь-яка програма, що розробляється як автоматизована інформаційна або керуюча система, вимагає вирішення багатьох, нерідко складних питань, таких як:

– вибору технічних засобів та операційного середовища;

- організації обчислювального процесу;

- створення структури та генерації БД;

- організації зберігання та відновлення даних;

– захисту даних від несанкціонованого доступу;

- організації інформаційного обміну (включаючи формат даних);

– обробки та запису базотворчих повідомлень;

- вибірки та читання даних;

- організації системи запитів;

- видачі діагностики;

– реєстрації обчислювальних подій та повідомлень у системі;

- забезпечення зав’язків з іншими (зовнішніми) системами.

В липні 2012 року була введена в експлуатацію Єдина автоматизована система керування вантажними перевезеннями (АСК ВП УЗ-Є) - замість систем АСОУП та АСК ВП УЗ, які функціонували на залізницях України.

Нова система архітектурно побудована для централізованого керування процесом вантажних перевезень.

Система АСК ВП УЗ-Є ґрунтується на електронному документообігу.

Функції складових частин автоматизованих робочих місць:

1) автоматизоване робоче місце чергового по станції (АРМ ДСП), який призначений для вирішення завдань, пов’язаних з технологічним процесом, прийому, відображення та зберігання інформації про положення на станції, ідентифікації і відстеження рухомих одиниць, оповіщення людей, що працюють на коліях. Автоматична система встановлюється безпосередньо на робочому місці чергового по станції (або оператора);

2) автоматизоване робоче місце оператора станційного технологічного центру (АРМ СТЦ) дозволяє автоматизувати такі процеси: складання сортувальних листків, безперервний номерний облік наявності і розташування вагонів на шляхах накопичення, підрахунок довжини і маси накопичувальних груп вагонів і складів, внесення змін до кількості і розташування вагонів на шляхах накопичення, складання натуральних листів на сформовані склади, підготовку і передачу в товарну контору і клієнтурі даних про очікуване прибуття вагонів під навантаження, підготовку та видачу довідок для заповнення маршруту машиніста, ведення форм обліку та звітності вагонного парку на станції;

3) автоматизоване робоче місце оператора пункту технічного огляду (АРМ ПТО) відображає технічний стан вагонів, контролює оформлення актів форми ВУ-36, ВУ-22;

4) автоматизоване робоче місце оператора пункту комерційного огляду (АРМ ПКО) призначене для працівників станції, зайнятих складанням актів загальної форми ГУ-23. Тут автоматизується вирішення таких завдань: складання актів загальної форми ГУ-23 на вагони з комерційними несправностями (оформлення, друк акта, корегування та повторна роздруківка, видалення складеного акта), видача оперативних повідомлень на складені акти загальної форми; отримання журналу реєстрації вагонів з комерційними несправностями форми ГУ-98;

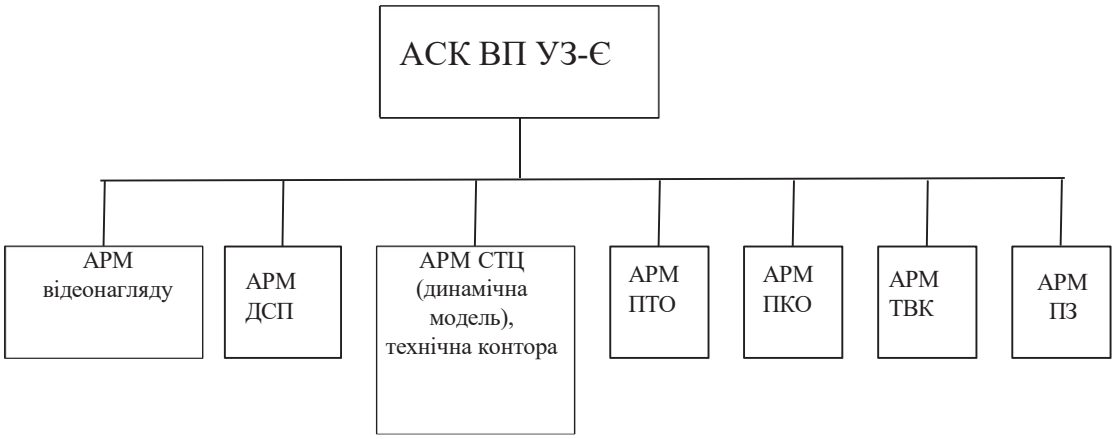

Рисунок 1.1 - Складові частини АСК ВП УЗ-Є

5) автоматизоване робоче місце товарного касира (АРМ ТВК) призначене для розрахунку провізної плати; автоматичного формування звітів ГУ-3, ГУ-5, ГО-1, ГО-2, ГО-3; формування та друку книги прибуття ГУ-42; ведення та друку накопичувальних карток; ведення особових рахунків клієнтів; друку платіжного доручення і формування картотеки платіжних доручень; введення конвенційних заборон (на прийом вантажу до перевезення, на прийом вантажу вантажоодержувачами, карантинна заборона тощо); ведення архіву відправок; ведення книги ГУ-34/ГУ-42;

6) автоматизоване робоче місце прийомоздавальника (АРМ ПЗ) є частиною комплексної системи «Автоматизована система обліку роботи під'їзних колій», яка призначена для ведення обліку інформації про фактичне місце дислокації вантажних вагонів, нарахування плати за користування вагонами та інших зборів, пов'язаних з користуванням вагонами на підприємствах. За допомогою АРМ ПЗ працівники станцій мають можливість обліковувати такі операції: подавати та забирати вагони на під'їзних коліях промислових підприємств та структурних підрозділів залізниці; на підставі введеної інформації про подавання та забирання вагонів нараховувати плату за користування вагонами, плату за подавання-забирання, а також збір за маневрову роботу; виконувати запит вагонних листів на відправлення [12, 13].

Впровадження АСК ВП УЗ-Є і перехід на нову, більш перспективну платформу дозволили:

- уніфікувати в масштабах АТ «Укрзалізниця» комплекс технічних засобів, забезпечити масштабованість усіх вузлів системи, суттєво розширити можливості з використання сучасних високоефективних стандартних програмних продуктів і технологій загальносистемного призначення;
- від окремих БД перейти до єдиної інтегрованої БД галузі, відповідним чином декомпозованої та розподіленої між вузлами глобальної мережі АТ «Укрзалізниця»;
- перейти на новий, сучасний стандарт інформаційного обміну в системі, максимально сумісний з Internet, автоматизованими системами інших відомств та держав;
- від численних включень обчислювальних операцій в технологічні процеси управління перевезеннями перейти до єдиної технології взаємодії користувачів із загальною БД, при якій участь будь-якої посадової особи у процесі управління може бути зведено до чіткої регламентації її персональної взаємодії з галузевою АСК;
- закласти орієнтацію на постійний розвиток системи в її фундаментальні принципи побудови і забезпечити її відкритість для створення нових завдань.

Основні переваги АСК ВП УЗ-Є над її попередниками:

- оперативність надходження інформації, яка веде за собою значну економію часу на переробку та аналіз документації;
- надійність безперебійного забезпечення даними. Це стало можливим після придбання та встановлення на базі Головного інформаційно-обчислювального центру в Києві найсучаснішого обладнання в цій галузі;
- економія фінансових коштів, яка досягається завдяки встановленню одного потужного комплексу замість шести окремих для кожної з регіональних філій-залізниць;
- можливість надання потрібної інформації не лише безпосереднім учасникам перевізного процесу (наприклад, службі перевезень) і галузевим господарствам, які забезпечують безпечність і надійність функціонування регіональних філій-

залізниць (служби локомотивного, колійного, енергетичного господарств та інших) при взаємодії з відправниками й одержувачами вантажів (клієнтами);

- наявність так званого «штучного інтелекту» системи, тобто можливість не тільки приймати і передавати інформацію, а й аналізувати, осмислювати, узагальнювати її та автоматично формувати довідки. Наприклад, якщо при прийманні вантажів до перевезення документи містять некоректні дані або не відповідають формі, система миттєво знаходить помилки і повертає документи на доопрацювання;
- можливість інтегрування вже розглянутих вище автоматизованих систем для швидкого і точного отримання потрібних даних.

1.2 Інформаційно-керуюча система розпуску складів (ГПЗП-М)

Однією з основних функцій, яка виконується на сортувальній гірці сортувальної станції залізничного транспорту - переміщення окремих вагонів, чи групи вагонів (відчепів) потягу, що прибув у парк прибуття СС, на одну з колій сортувального парку, де формуються нові потяги.

Дана операція реалізується шляхом відчеплення відчепу від составу, що розформовується на вершині сортувальної гірки та вільного скочування цих ввідчепів на колію СП по сформованому на спуску з гірки шляху.

ГПЗП дозволяє здійснювати повне розформування поїздів за заданою програмою, звільняючи оператора від завдання різних команд; одночасно забезпечується автоматична видача оператору, розчепщіку та іншим працівникам гірки низки сигналів про хід розпуску.

Для автоматизації ряду елементів цієї операцій розроблений гірковий програмно-заїдаючий пристрій (ГПЗП), який являє собою програмно-апаратний комплекс, основу якого складає сучасний контролер. Загальна структура комплексу представлена на рис. 1.2.

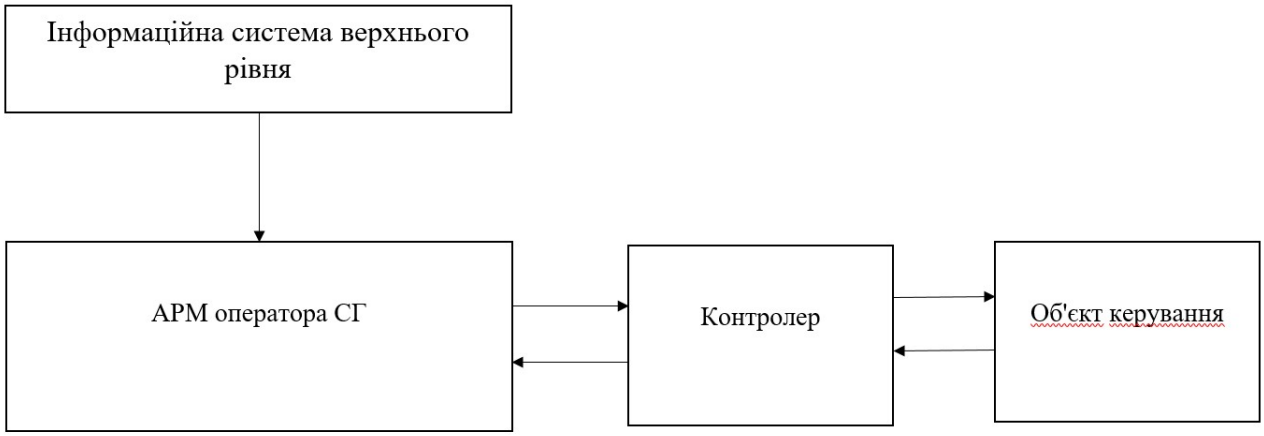

Рисунок 1.2 - Загальна структура комплексу ГПЗП-М

Розроблений програмно-апаратний комплекс реалізує наступні функції:

- виконуються на Армі оператора: прийом сортувального листа по каналах зв'язку від системи верхнього рівня;
- виконуються на Армі оператора: підготовка даних (програми розпуску) для контролера;
- виконує контролер: видача на колійні індикатори на вершині гіркі значень про кількість вагонів у двох чергових відцепах;
- виконує контролер: завантаження маршрутного накопичувача ГАЦ даними про шляхи проходження для кожного з відцепів, що скачуються.

АРМ оператора отримує інформацію від системи верхнього рівня з використанням каналів зв’язку. Це дані про стан парку прибуття та маршрути розпуску составів, при цьому використовується стандартна програма розсилання повідомлень на сортувальних станціях (програма «Богатопротокольний маршрутизатор»).

АРМ і контролер час від часу виконують обмін повідомленнями між собою. Повідомлення є службові та інформаційні. Службові повідомлення призначені для організації передачі інформаційних повідомлень і відстеження вірної роботи системи. Інформаційні повідомлення це інформація про події, що відбуваються в процесі розпуску або зміною ситуації в парку прибуття. Для організації обміну даними був розроблений спеціальній протокол взаємодії. Зв'язок здійснюється з використанням LAN портів.

Основні функції АРМу:

- організація зв'язку з системою верхнього рівня з метою отримання поточної інформації в парку прибуття та сортувальних листів;
- організація зв'язку з контролером з метою отримання інформації про об’єкт управління та керування процесом скочування відчепів;
- відображення інформації готових до розпуску складів у парку прибуття, програм розпуску складів, поточної інформації о процесе скочування відчепів та надання можливості коригувати програми розпуску складів.

По мірі готовності составів до розформування - сортувальні листки для составів що знаходяться в парку прибуття, пересилаються з інформаційного центра в базу даних АРМу. Оператор гіркі обирає состав, який прибуває на вершину гірки та корегує (за необхідністю) сортувальний лист цього составу, В процесі розпуску, після відчеплення наступного відчепу від составу контролер ГПЗП за сигналами з колійних датчиків пересилає закодовасоставні повідомлення щодо цього відчепу з сортувального листка в маршрутний накопичувач. Також одночасно інформація про наступний відчеп (що потребує відчеплення) пересилається контролером на колійний індикатор для інформування про кількість вагонів, що потрібно відчепити.

1.3 Висновки

Створення та впровадження системи АСК ВП УЗ, та згодом систему АСК ВП УЗ-Є забезпечило незалежність автоматизованих систем на залізницях України. Універсальні базові принципи та методи ціх систем, орієнтовані на інтеграцію корпоративних АСК та перехід до об'єктно-орієнтованих методів моделювання процесів перевезення, відкривають перспективи для перетворення автоматизованих систем на інтелектуальні системи підтримки прийняття рішень персоналом на різних рівнях. За підсумками принципів АСК ВП УЗ забезпечується глобальна інформатизація перевізного процесу, насамперед це з створенням мережевої моделі управління вантажними перевезеннями.

Проведено аналіз застосування єдиної інформаційної системи. Вивчено вплив автоматизованої системи керування вантажними перевезеннями на

підвищення достовірності інформації через скасування обміну даними між залізницями - завдяки концентрації інформації в єдиній інформаційній базі.

Розробка інформаційної взаємодії автоматизованих робочих місць операторів сортувальних станцій при виконанні технологічних операцій має забезпечити прискорення розформування та формування составів, прискорити обіг вагонів на сортувальній станції.

2 ПОРІВНЯЛЬНИЙ АНАЛІЗ ПРОГРАМНОГО ТА АПАРАТНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Вибір програмного середовища реалізації проекту

Для розробки ПЗ АРМу мною обрана система об’єктно-орієнтовного програмування Delphi 7.0., тому що вона в безкоштовному доступі на даний час.

Розробка складних програмних продуктів, оформлених відповідно до сучасних вимог, практично неможлива без використання інструментальних засобів швидкої розробки програм - одним з таких засобів є система програмування Delphi 7, створена корпорацією Borland і призначена для розробки програм з використанням середовища візуального проектування.

Як і у будь-якому іншому середовищі візуального програмування, у середовищі Delphi процес розробки програми складається з двох етапів, перший з яких орієнтований на роботу з візуальними конструкторами, а другий – на роботу з програмним кодом. Характерним при цьому є те, що в процесі виконання другого етапу (безпосереднього програмування) завжди можна повернутися до першого з метою корекції проекту та наступним продовженням написання програмного коду. Ця особливість забезпечує гнучкість процесу розробки програми.

Візуальним відображенням працюючої програми, що написана у Delphi, є форма, яку і розробляє програміст, починаючи проектування програми. Спеціальне вікно форми створюється автоматично, як тільки програміст у візуальному середовищі вибере команду Application (Додаток). При цьому середовище Delphi автоматично створює оформлений за певними правилами спеціальний файл опису форми, що має розширення .DFM. Крім того, створюється пов’язаний з формою та оформлений спеціальним чином файл, що називається модулем форми.

Перший етап проектування програми полягає в наповненні вікна форми елементами керування, які називаються компонентами. Такими елементами є різні кнопки, прапорці, списки, вікна редакторів тощо. Програміст оперує компонентами, маючи їхнє візуальне відображення. Можна сказати, що він

просто бере їх з так званої палітри компонентів і розміщає на формі.

Якщо програміст поміщає на форму який-небудь компонент, то Delphi автоматично робить відповідні зміни у файлі форми, а також у модулі, вносячи в останній посилання на доданий компонент. При цьому навіть порожній, створеній автоматично формі відповідає програма, що може бути виконана після компіляції.

Компоненти, що входять у бібліотеку візуальних компонентів Delphi, фактично є програмними заготівками для майбутньої програми, оскільки вони містять у собі як програмний код, так і дані, необхідні для його функціонування. Тому і програма в цілому, і розміщені у вікні форми компоненти є працездатними навіть без зміни автоматично створеного програмного коду. Якщо автоматично встановлені налаштування яких-небудь із розміщених на формі компонентів не влаштовують програміста, він може ввести зміни без звертання до програмного коду. Для цього в Delphi передбачено засоби, що дозволяють змінювати характеристики компонентів за допомогою мишки або клавіатури вже на етапі проектування.

На другому етапі розробки програми програміст повинен наповнити компоненти програмним кодом, призначеним для розв’язання поставленої задачі. При цьому, природно, йому багато в чому доводиться орієнтуватися на традиційну техніку програмування. У той же час вживання компонентів істотно змінює техніку програмування: сучасне програмування базується на об’єктно-орієнтованому підході з використанням таких понять, як класи, властивості, методи та події. Установлені при проектуванні форми початкові налаштування компонентів можуть змінюватися програмно в процесі виконання додатка. Більш того, у процесі виконання програми компоненти можуть програмно створюватися та знищуватися.

У процесі написання програмного коду програміст завжди може повернутися до етапу проектування, здійснюючи модифікацію форми в плані не тільки її візуального відображення, але і наповненості компонентами.

Достатньо потужні засоби налаштування дозволяють виконувати програму за

операторами, стежачи за ходом виконання по тексту і контролюючи при цьому поточні значення змінних. Можлива також установка точок припинення (переривання), при досягненні яких програма автоматично переривається свою роботу, переходячи у налаштовуваний режим.

Початкові відомості про середовище розроблювача Delphi 7.

Інтегроване середовище розроблювача (ICP) Delphi візуально реалізуються декількома одночасно відкритими вікнами (рис. 2.1), які програміст може переміщувати по екрану, створюючи собі комфортні умови для роботи.

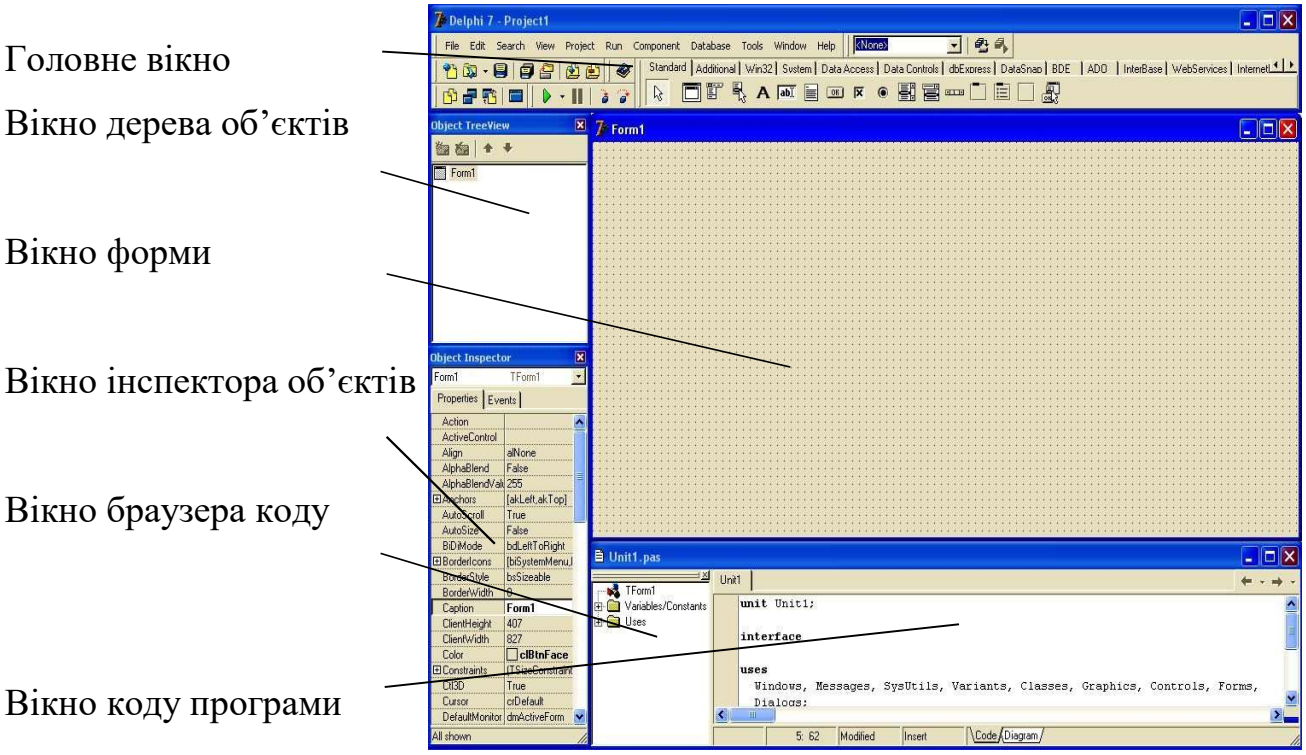

Рисунок 2.1 - Найважливіші вікна ICP Delphi

Вікна можна розміщати таким чином, щоб вони частково перекривали або повністю закривали одне одного зі збереженням можливості доступу до них. При цьому кожне вікно призначене для вирішення цілком конкретного завдання. Кожне з вікон, крім головного, може бути закрите. Закриття ж головного вікна приводить до виходу з ICP Delphi.

Головне вікно.

Це вікно містить у собі всі засоби з керування проектом:

головне меню, набір інструментальних кнопок і палітру компонентів.

Усі елементи головного вікна згруповані на окремих панелях, які мають спеціальні вішки, призначені для переміщення цих панелей при їхньому захоплюванні мишкою і перетаскуванні при натиснутій її лівій кнопці. Такі ж вішки з'являються й у вікон при вбудовуванні їх усередину інших вікон. Будь-яку панель головного вікна (крім головного меню) можна прибрати з екрана.

За допомогою головного меню здійснюється керування можливостями ICP. Швидкий доступ до багатьох команд головного меню забезпечують так звані інструментальні кнопки, згруповані за функціональним призначенням на окремих панелях головного вікна.

Панель Стандартна містить інструментальні кнопки, що забезпечують дублювання тих дій, які можна виконати, звернувшись до пунктів Файл і Проект головного меню.

Панель Вигляд поєднує кнопки, що дублюють підпункти пункту View (Вигляд) головного меню, а панель налаштування полегшує керування процесом налаштування та виконання програми, частково дублюючи підпункти пункту Run (Виконати) головного меню.

Палітра користувача за умовчанням вміщує інструментальну кнопку, що забезпечує доступ до довідкової системи Delphi.

Особливе місце займає Палітра компонентів – багатосторінкова панель, на сторінках якої розташовуються інструментальні кнопки, що забезпечують додавання компонентів до активної форми при проектуванні інтерфейсу програми.

Вікно форми

Вікно форми (конструктор форми) являє собою проект вікна створюваної програми. Спочатку воно містить кнопки виклику системного меню, розгортання, згортання і закриття вікна, рядок заголовка і габаритну рамку (ці елементи є стандартними інтерфейсними елементами Windows), а також порожню робочу область, звичайно заповнену крапками координатної сітки, що служить для полегшення позиціювання розташовуваних на формі компонентів і відображається тільки на етапі конструювання програми.

За допомогою команди Tools ► Environment Options (Сервіс ►Опції Середовища) можна викликати вікно налаштувань середовища і, знявши прапорець Display Grid (Показувати) на вкладці (сторінці) Designer (Дизайнер), прибрати координатну сітку з форми.

Будучи візуальним засобом розміщення компонентів на формі, конструктор форми дозволяє безпосередньо за допомогою миші розміщати компоненти на формі з використанням палітри компонентів головного вікна, Інспектора Об'єктів (Object Inspector) або Дерева Об'єктів (Object Tree).

Вікно дерева об’єктів

У вікні Дерева Об’єктів (Object TreeView) наочно (у вигляді дерева) відображаються зв’язки між окремими візуальними та невізуальними компонентами, розміщеними на активній формі або в активному модулі даних, а саме, приналежність одних компонентів іншим. Окремі компоненти відображаються в Дереві Об’єктів за допомогою піктограм (маленьких рисунків), поруч із якими містяться імена відповідних компонентів. Клацання (клік) лівою кнопкою мишки на кожній з таких піктограм приводить до активізації відповідного компонента у вікні форми і відображення його властивостей у вікні Інспектора Об’єктів (Object Inspector). При подвійному кліку на піктограмі компонента спрацьовує так званий механізм Code Insight, що вставляє у вікно коду заготівку для оброблювача події OnClick (За кліком) або оброблювача іншої події (залежно від типу компонента).

Використовуючи Дерево Об’єктів, можна змінити власника компонента, для чого досить клацнути лівою кнопкою мишки над піктограмою і, не відпускаючи кнопку мишки, «перетягнути» її у вікні на піктограму нового власника. Натискання клавіші Delete приводить до видалення компонента, який виділений у Дерево Об’єктів.

Вікно інспектора об’єктів

Вікно Інспектора Об’єктів (Object Inspector) призначене для зміни параметрів (характеристик) розміщених на формі компонентів. За допомогою мишки у вікні форми можна коректувати тільки частину параметрів, які характеризують

компоненти, що розміщені на формі. Користуючись же Інспектором Об’єктів, програміст може коректувати всі характеристики компонентів, які доступні на етапі проектування програми (наприклад, колір, шрифт і текст напису, колір компонента). Для здійснення цих змін в Інспекторі Об’єктів використовуються дві вкладки (сторінки) – Properties (Властивості) та Events (Події). За допомогою вкладки Properties (Властивості) програміст може змінювати властивості компонента, а за допомогою вкладки Events (Події) – призначати реакцію компонента на ту або іншу подію (реакцію на натискання клавіш, клік мишкою, зміну розміру вікна, появу компонента на екрані тощо).

Для зміни властивостей компонента необхідно перейти в Інспекторі Об’єктів на першу вкладку, що має вигляд таблиці, у першому стовпчику якої містяться імена властивостей. Значення властивості відображається поруч з її ім’ям у другому стовпчику.

Властивості компонентів можуть відображатися єдиним значенням (числом, рядком символів, True – Істина чи False – Неправда) або множиною значень. У першому випадку говорять про прості властивості, а в другому – про складні. Ліворуч від імені складної властивості відображається значок «+». Для задавання значення простої властивості необхідно вибрати відповідний рядок і ввести потрібне значення в правому стовпчику. Якщо при виборі простої властивості в правому кінці рядка з’являється кнопка  , то це означає, що для даної властивості визначений список можливих значень, який розкривається при кліку над кнопкою  і служить для подальшого вибору потрібного значення. Для зміни значення складної властивості слід клацнути мишкою над значком «+» поруч з його ім’ям, у результаті чого відкривається список складових цієї властивості. Закриття цього списку здійснюється кліком мишкою над значком «–», у який перетворюється значок «+» при відкритті списку.

Наприкінці рядка для деяких із властивостей при їх активізації може з’явитися кнопка  . Клік над цією кнопкою приводить до появи на екрані діалогового вікна, що служить для установки значення властивості.

Вкладка Events (Події) також має вигляд таблиці з декількох рядків і двох

стовпчиків. У першому стовпчику відображається ім’я події, а в другий – ім’я підпрограми для її опрацювання.

Крім двох вкладок, у вікні Інспектора Об’єктів є список, який може розкриватися і містить імена всіх поміщених на форму компонентів із зазначенням їхніх класів. При кліку мишкою над ім’ям компонента в цьому списку відбувається переключення на відповідні таблиці в нижній частині Інспектора Об’єктів, а також активізація обраного компонента у вікні форми, що позначається виділенням компонента прямокутником.

Налаштування вікна Інспектора Об’єктів можна виконати за допомогою контекстного меню, яке відкривається при кліку правою кнопкою миші.

Вікно коду програми

У вікні коду програми відображається і редагується текст програми (найчастіше текст модуля), написаної мовою Delphi, базою якої є алгоритмічна мова Object Pascal. При розробці нового проекту це вікно містить створюваний автоматично мінімальний початковий текст модуля. Цей текст забезпечує нормальне функціонування порожньої форми. Далі програміст модифікує даний текст відповідно до розв’язуваної задачі, причому зміни виконуються як за допомогою засобів VCL, так і за допомогою «ручної» корекції. При цьому досить часто доводиться переходити з вікна коду у вікно форми і зворотно. Якщо потрібне вікно хоча б частково видно на екрані, перехід може бути здійснений кліком мишкою над видимою частиною вікна.

Якщо вікно коду закрите вікном форми (або навпаки), то для переходу необхідно клацнути над інструментальною кнопкою  , розташованою на панелі Вигляд ICP Delphi або натиснути клавішу <F12>.

Вікно браузера коду

Вікно Браузера Коду (Code Explorer) звичайно активізується разом з вікном коду і служить для полегшення пошуку потрібних елементів у великому тексті програми. Вікно браузера коду містить елементи, що використовуються в програмі.

При подвійному кліку мишею над елементом у вікні браузера коду екстовий

курсор у вікні коду автоматично переміщається на опис цього елемента або на ядок, у якому він згадується вперше. За допомогою браузера коду можна також проводити додавання і перейменування елементів програми.

Горизонтальна вішка у верхній частині вікна браузера коду дозволяє еретаскувати його за допомогою мишки, розташовуючи в будь-якому зручному для програміста місці екрана.

Закрити вікно браузера коду можна, клікнувши мишкою над кнопкою в його правому верхньому куті. Для відкриття раніше закритого вікна браузера коду треба або клацнути правою кнопкою мишки над вікном коду програми і вибрати опцію View Explorer (Показати Огляд), або скористатися меню, виконавши команду View ► Explorer (Показати ►Огляд Коду).

2.2 Вибір сервера бази даних

Для роботи з БД в проекті я обрав систему керування базами даних (СКБД) InterBase 6.5 тому що вона є в безкоштовному доступі на даний час.

Короткий опис InterBase 6.5.

InterBase SQL Server - це система керування базами даних спочатку розроблена компанією Borland. В даний час розробником InterBase є Embarcadero.

Версійна архітектура InterBase SQL Server.

База даних InterBase побудована на версійній архітектурі зберігання даних.

Цей підхід має ряд переваг перед блокувальними СКБД:

• для відновлення баз даних InterBase після системного збою немає необхідності підтримки лога транзакцій;

• клієнти, які читають дані, ніколи не блокують клієнтів, які здійснюють запис даних.

Переваги сервера InterBase.

Сервер InterBase - це кросплатформова СКБД, що підтримує більшість операційних систем: Windows, Linux, Unix, Solaris, Mac OS і т.д.

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

• поновлення представлення View;

• двофазне підтвердження транзакцій;

• ефективний механізм тригерів;

• серверна обробка BLOB-полів (BLOB-filters);

• події (повідомлення);

• шифрування мережного трафіку, бази даних, файлів бекапа та окремих стовпців БД.

Мова InterBase SQL сумісна з стандартом SQL-92. Крім того, InterBase server підтримує розширення стандартного підмножини мови SQL за рахунок функцій користувача UDF (User Definded Functions). InterBase SQL надає розширені можливості SQL для збережених процедур і тригерів - PSQL.

Продуктивність InterBase.

Сервер InterBase розроблявся з урахуванням сучасних вимог щодо продуктивності СКБД. В останніх версіях InterBase SQL Server застосовано низку технологій, що значно підвищують швидкість роботи:

• підтримка симетричної багатопроцесорної обробки InterBase SMP (symmetric multiprocessing) дозволяє використовувати можливості багатопроцесорної архітектури при єдиному серверному процесі InterBase SuperServer;

• підтримка технології Hyperthreading;

• пакетне виконання SQL запитів дозволяє зменшити мережевий трафік та підвищити продуктивність.

Створення нової бази даних InterBase.

Створити та підключити нову базу даних можна через консоль керування IBConsole.

У порожній базі даних необхідно створити таблиці, зв'язки між таблицями, первинні ключі, індекси, процедури, що зберігаються, генератори та інші об'єкти. Для редагування баз даних можна використовувати вбудовані засоби InterBase — IBConsole або утиліту isql.exe.

Скріншоти екранів представлені нижче.

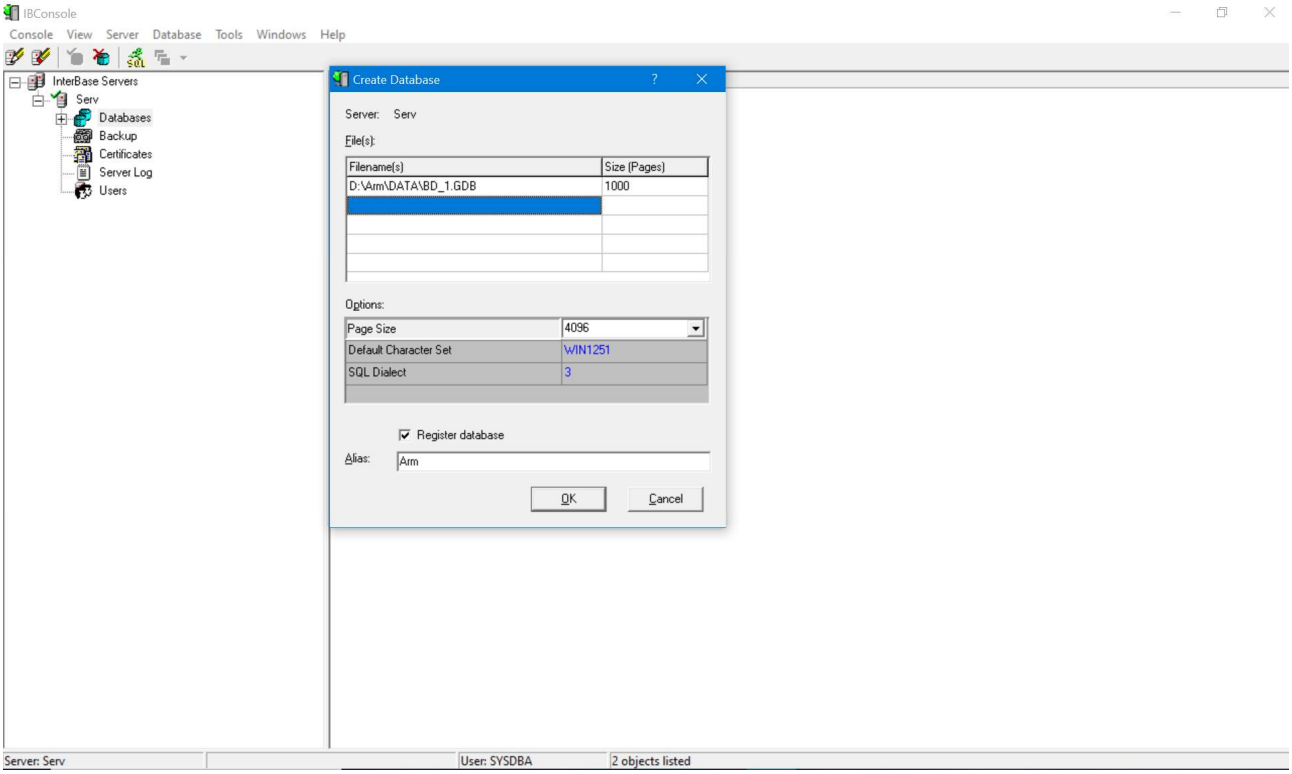

Рисунок 2.3 - Створення файлу БД BD_1.GDB

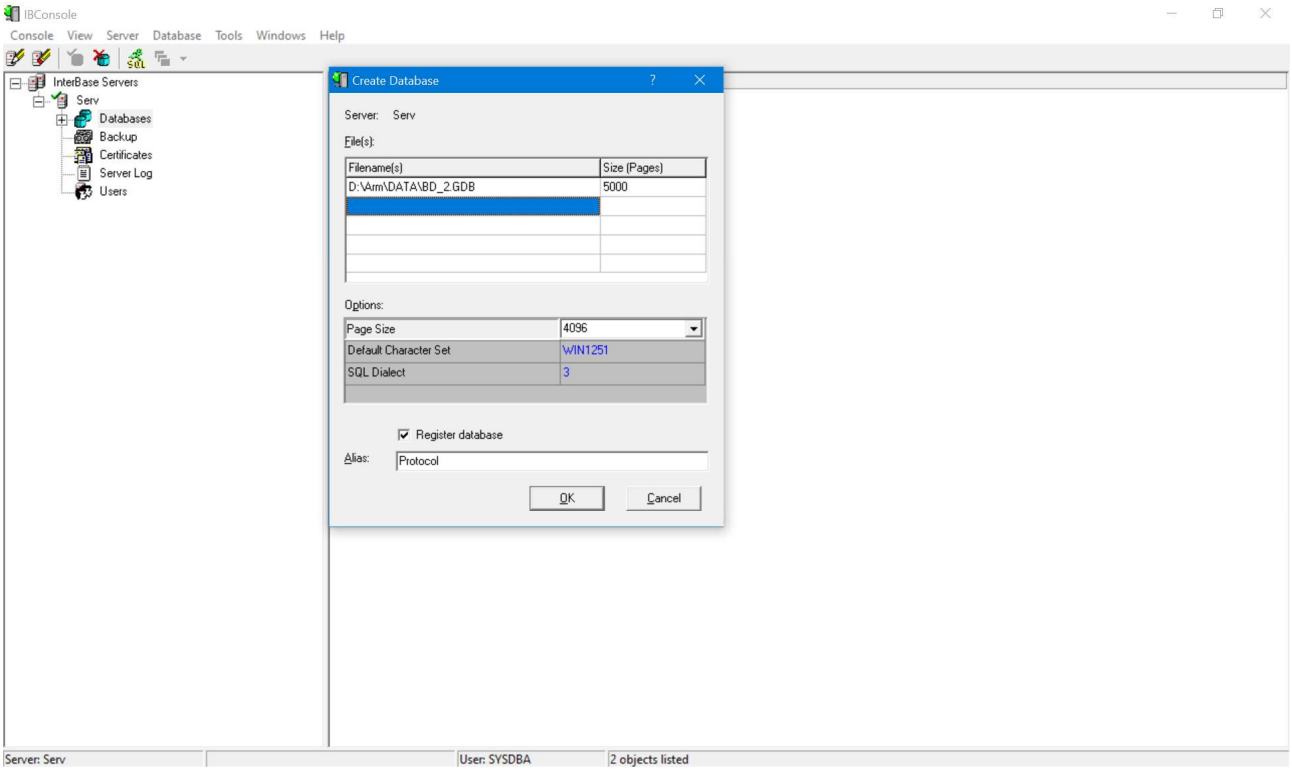

Рисунок 2.4 Створення файлу БД BD_2.GDB

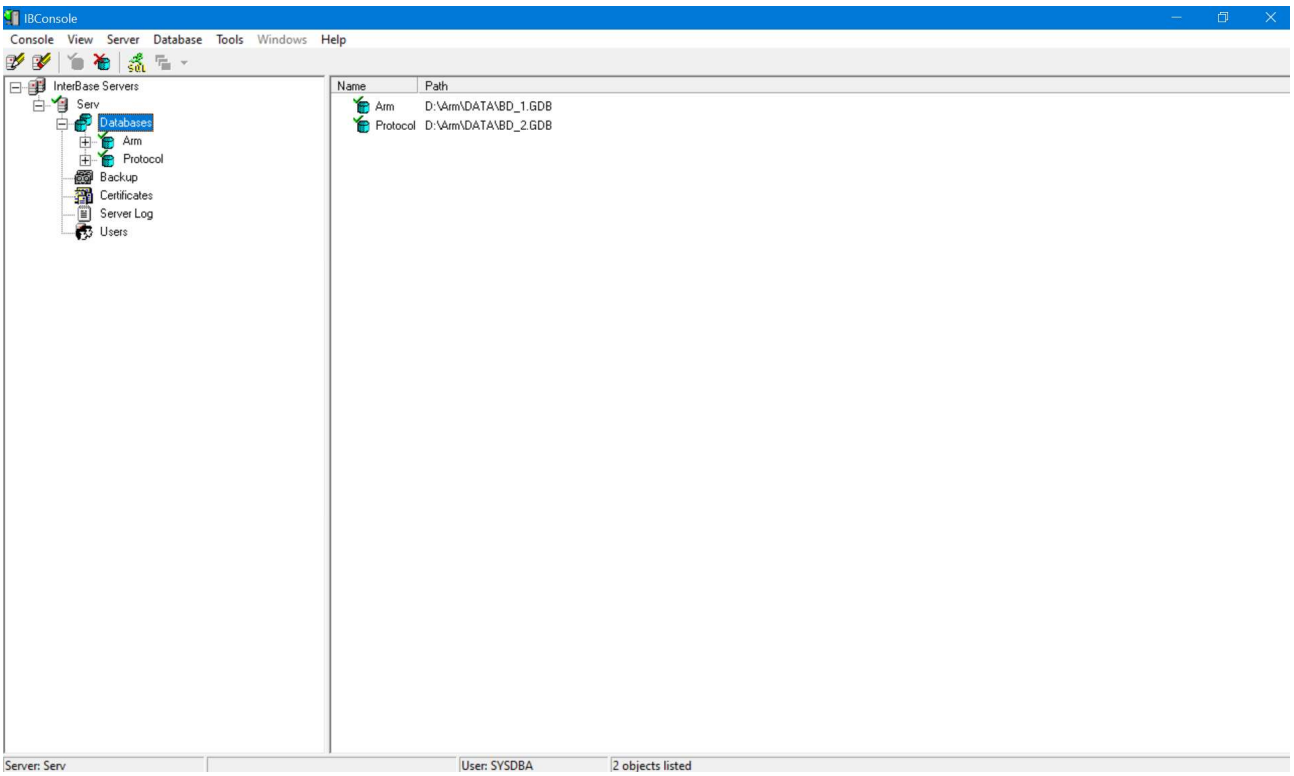

Рисунок 2.5 Підключення БД

Безпека InterBase Server

InterBase SQL Server підтримує кілька методів автентифікації користувачів:

• Classic Authentication Scheme – користувачі та паролі є єдиними для всіх баз і зберігаються в системній базі даних InterBase – admin.ib (isc4.gdb).

• Embedded User Authentication — користувачі та паролі зберігаються у базі даних клієнта. Така схема захищає бази даних від прямого. Обидві схеми аутентифікації Classic та EUA можуть бути використані на сервері InterBase спільно.

Існує кілька способів роботи з InterBase з Delphi:

• dbGo (ADO Express) компоненти, які працюють через бібліотеку ADO;

• прямий доступ до COM-інтерфейсів ADO, минаючи компоненти dbGo;

• прямий доступ до COM-інтерфейсів OLE DB за допомогою сторонніх VCL-компонентів (OLE DB Direct/OLE DB Express).

2.3. Рекомендації щодо вибору апаратного забезпечення

Для забезпечення ефективної роботи автоматизованого робочого місця (АРМ) оператора сортувальної гірки на сортувальній станції укрзалізниці

необхідно використовувати персональний комп’ютер (ПК), що відповідає наступним мінімальним вимогам.

Технічні параметри ПК:

1. Процесор: не нижче Intel Core i3 (10-го покоління) або AMD Ryzen 5 (аналогічної продуктивності).
2. Оперативна пам’ять: 4 ГБ або більше, рекомендується 8 ГБ для роботи з кількома додатками.
3. Жорсткий диск/SSD:

- накопичувач SSD на 256 ГБ для операційної системи та програмного забезпечення;

- додатково може бути предбачено SSD на 1 ТБ для зберігання даних (н априклад, статистики роботи сортувальної гірки, журналів, відеофіксації).

4. Графічний адаптер: вбудований графічний процесор, наприклад Intel UHD Graphics, достатній для офісних додатків. Якщо використовуються спеціалізовані програми з графічними елементами (наприклад, системи візуалізації або симуляції), рекомендується дискретна відеокарта NVIDIA GeForce GTX 1650 або вище.

5. Порти:

- мінімум 4 порти USB (2 порти USB 3.0 та 2 порти USB 2.0);
- порт HDMI або DisplayPort для підключення монітора;
- Ethernet-порт для підключения до локальної мережі;
- аудіовиход (3,5 мм) для підключення гарнітури або колонок.

6. Монітор: Розмір екрана 19–24 дюйми, роздільна здатність Full HD (1920×1080).

Рисунок 2.6 - Персональний комп’ютер оператора сортувальної гірки

Програмне забезпечення.

Для коректної роботи АРМу необхідно встановити наступне програмне забезпечення:

1. Операційна система: Microsoft Windows 10/11 Pro (ліцензійна версія).
2. Офісні програми: Пакет Microsoft Office (або аналог, наприклад, LibreOffice).
3. Спеціалізоване ПЗ:
   - АРМ оператора сортувальної гірки;
   - система візуалізації роботи гірки (наприклад, SCADA);
   - система збору та аналізу даних.
4. Додаткове обладнання:

- клавіатура та миша - дротові або бездротові пристрої;
- джерела безперебійного живлення (ДБЖ) для захисту обладнання від перебоїв в електропостачанні;
- принтер або БФП для друку звітів та інших документів.

Застосування вищевказаних рекомендацій дозволить забезпечити стабільну та безпечну роботу АРМа оператора сортувальної гірки, підвищивши ефективність та надійність виконання виробничих завдань.

2.4 Висновки

Для стоверення програмної частини системи «АРМ оператора сортувальної гірки» я обрав систему об’єктно-орієнтовного програмування Delphi 7.0., тому що вона доступна, має не складний інтерфейс та є в безкоштовному доступі на даний час.

Для роботи з БД в проекті я обрав систему керування базами даних (СКБД) InterBase 6.5.

Також для забезпечення ефективної (без додаткових втручань системних адміністраторів) роботи автоматизованого робочого місця (АРМ) оператора сортувальної гірки на сортувальній станції укрзалізниці необхідно використовувати персональний комп’ютер (ПК), що відповідає вимогам вказаним вище.

3 ОПИС ТЕХНОЛОГІЧНИХ ОПЕРАЦІЙ ТА ІНФОРМАЦІЙНИХ ПОТОКІВ АРМУ ОПРЕРАТОРА

3.1 Опис функціональних можливостей АРМу

Автоматизоване робоче місце чергового по гірці призначено для відображення оперативної інформації про ситуації на коліях парку прибуття, відповідно до готовності поїздів до процесу розпуску, а також можливості оператора змінювати програму розпуску поїздів з наступною ініціалізацією процесу розпуску.

Основні цілі і задачі АРМу:

– формування запиту в АСК ВП УЗ з метою одержання сортувального листа поїзда в парку прибуття готового до розпуску. Даний запит формується автоматично АРМом із заздалегідь заданим інтервалом;

– відображення інформації готових до розпуску поїздів у парку прибуття. Відображається інформація про поїзди, що знаходяться на коліях парку прибуття, для яких був отриманий сортувальний лист із АСК ВП УЗ. При цьому відображається безпосередньо сортувальний лист і програма розпуску поїзда;

– автоматичне відновлення інформації АРМу при надходженні сортувального листа або закінчення розпуску поїзда;

– здійснення обміну інформаційними і керуючими повідомленнями з контролером з метою керування контролером і спостереженням за справністю роботи системи в цілому. Виконується автоматично з заданим інтервалом часу;

– надання операторові можливості зміни програми розпуску поїзда. Виконання операцій дроблення й об'єднання відчепів з наступною зміною маршрутів;

– контроль правильності дій оператора. Заборона повторної або нової передачі програми розпуску в контролер у випадку не завершення розпуску поїзда для попередньої програми;

– здійснення операції пересилання програми розпуску в контролер, иконується з ініціативи оператора;

– відображення аварійної ситуації у випадку виходу з ладу контролера;

– можливість створення звітів сортувального листа і програми розпуску (друк на принтері).

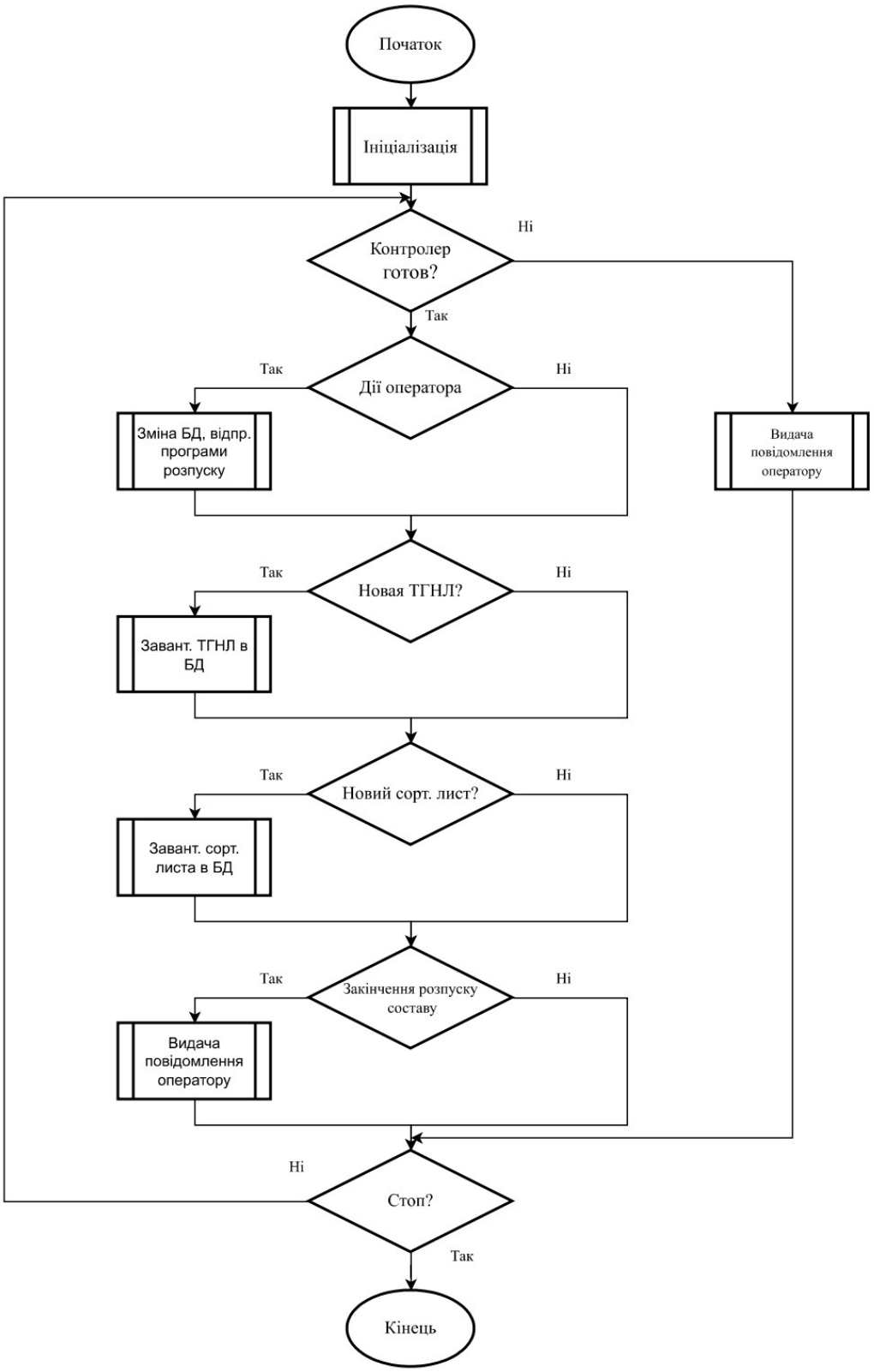

Рисунок 3.1 – Загальний алгоритм роботи АРМу оператора СГ

3.2. Організація інформаційного забезпечення

У процесі роботи АРМу відбуваються операції обміну інформацією з АСК ВП УЗ і контролером, відображення і збереження інформації про ситуації в парку прибуття, задачах та процесі розпуску поїздів. Для організації виконання цих задач були розроблені протоколи обміну між контролером і АРМом, структури таблиць бази даних. Для організації обміну даними з АСК ВП УЗ були прийняті стандартні протоколи, застосовувані на сортувальних станціях.

3.2.1 Структура бази даних

Для створення серверу бази даних мною була вибрана СКБД InterBase 6.5.

База даних розроблена та представлена у вигляді чотирьох таблиць:

– TrainPP (інформація про ситуації в парку прибуття);
– SortList (сортувальний лист поїзда);
– ProgrRospusk (програма розпуску поїзда);
– ExchangeData (повідомлення обміну даними з контролером і АСК ВП УЗ).

Таблиця TrainPP містить інформацію про поїзди, що знаходяться на коліях парку прибуття і готових до процесу розпуску. Дані про такі поїзди потрапляють у таблицю після одержання сортувального листа з АСК ВП УЗ. Видалення даних з таблиці TrainPP відбувається при надходженні сигналу з контролера про закінчення процесу розпуску.

Таблиця SortList містить інформацію сортувального листа поїзда готового до процесу розпуску, отриману з АСК ВП УЗ.

Таблиця ProgrRospusk містить інформацію про програму розпуску поїзда. Спочатку дані таблиць ProgrRospusk і SortList однакові. Однак, оператор має можливість змінювати дані в таблиці ProgrRospusk до того як програма розпуску буде передана в контролер.

Таблиця ExchangeData призначена для збереження всіх повідомлень обміну інформацією між АРМом, контролером і АСК ВП УЗ. Ця таблиця призначена для службового користування.

Структури цих таблиць бази даних представлені в таблицях 3.1-3.4. SQL-файл бази даних, що містить лістинг програми, який створює використовувані таблиці, індекси, генератори і тригери, представлений у додатку А.

Таблиця 3.1 - Структура таблиці TrainPP (інформація про ситуації в парку прибуття)

№ п\п	Ім'я поля	Тип поля	Опис поля
1.	TrainPP_Id	INTEGER	Первинний ключ
2.	DT_InputSL	DATE	Дата і час одержання сортувального листа
3.	NumWay	CHAR(2)	Номер колії в парку прибуття
4.	NumTrain	CHAR(4)	Номер поїзда
5.	InfTrain	CHAR(13)	Додаткова інформація про поїзд: станція формування, порядковий номер і станція призначення
6.	WeightTrain	CHAR(5)	Вага нетто поїзда
7.	LehgthTrain	CHAR(4)	Умовна довжина поїзда
8.	DT_EndRospusk	DATE	Дата і час закінчення розпуску

Таблиця 3.2 - Структура таблиці SortList (сортувальний лист поїзда)

№ п\п	Ім'я поля	Тип поля	Опис поля
1.	SortList _Id	INTEGER	Первинний ключ
2.	TrainPP_Id	INTEGER	Поле зв'язку з таблицею TrainPP
3.	NumOtcepa	INTEGER	Номер відчепа
4.	NumWay	CHAR(2)	Номер колії, маршрут
5.	QuanCars	INTEGER	Кількість вагонів у відчепі
6.	CheckRol	CHAR(1)	Оцінка про ролики
7.	WeightOtcepa	CHAR(4)	Вага нетто відчепа
8.	NumFirstCar	CHAR(8)	Номер першого вагона у відчепі

Таблиця 3.3 - Структура таблиці ProgrRospusk (програма розпуску поїзда)

№ п\п	Ім'я поля	Тип поля	Опис поля
1.	ProgrRospusk _Id	INTEGER	Первинний ключ
2.	TrainPP_Id	INTEGER	Поле зв'язку з таблицею TrainPP
3.	NumOtcepa	INTEGER	Номер відчепа
4.	NumWay	CHAR(2)	Номер шляху, маршрут
5.	QuanCars	INTEGER	Кількість вагонів у відчепі
6.	CheckRol	CHAR(1)	Оцінка про ролики

Таблиця 3.4 - Структура таблиці ExchangeData (повідомлення обміну даними з контролером і АСК ВП УЗ)

№ п\п	Ім'я поля	Тип поля	Опис поля
1.	ExchangeData _Id	INTEGER	Первинний ключ
2.	DT_Exchange	DATE	Дата і час одержання\відправлення повідомлення
3.	In_Data	CHAR(1)	Прийом повідомлення (‘A’- від контролера, ‘B’- від АСК ВП УЗ)
4.	Out_Data	CHAR(1)	Відправлення повідомлення (‘A’- у контролер, ‘B'-в АСК ВП УЗ)
5.	Data_Exchange	BLOB	Вміст повідомлення
6.	Remar	VARCHAR(100)	Примітка

3.2.2 Структура повідомлень обміну інформацією між АРМом і контролером

АРМ і контролер у процесі роботи виконують обмін повідомленнями між собою. Ці повідомлення можна представити як службові й інформаційні.

Службові повідомлення призначені для організації передачі інформаційних повідомлень і відстеження коректної роботи системи.

Інформаційні повідомлення зв'язані з передачею інформації про події, що відбуваються в процесі розпуску або зміною ситуації в парку прибуття. У таблиці 3.1 представлені види повідомлень, якими обмінюються АРМ і контролер.

Таблиця 3.1 Перелік повідомлень, якими обмінюються АРМ і контролер

Повідомлення, передані з АРМа в контролер		
Назва повідомлення	**Формат повідомлення**	**Опис**
1	2	3
Програма розпуску	(:01 хххх ннн мм вв:)	Програма розпуску передається в контролер у виді набору повідомлень. Кожне повідомлення включає поля: хххх - номер потяга; ннн - порядковий номер відчепу; мм - маршрут відчепу; вв - кількість вагонів у відчепі. Останнє повідомлення програми розпуску має нульове поле "вв". Вимагає відповіді "Зрозумів" чи "Не зрозумів, повторити!"
Запит готовності прийому даних	(:02:)	Повідомлення передається, як правило, перед програмою розпуску. Вимагає відповіді "Готовий" чи "Не готовий"
Зрозумів	(:10:)	Відповідь на ряд повідомлень контролера у випадку їхнього

		правильного формату при верифікації.
Не зрозумів, повторити!	(:09:)	Відповідь на ряд повідомлень контролера у випадку їхнього неправильного формату при верифікації.
Повідомлення, передані з контролера в АРМ		
Готовий!	(:02:)	Відповідь на запит про готовність контролера до прийому програми розпуску. Формується у випадку, якщо пристрій працює в автоматичному режимі (натиснута кнопка "Авт/Руч" на пульті оператора і рукоятка перемикача режимів встановлена в положення "Програмний"). Вимагає відповіді "Зрозумів" чи "Не зрозумів, повторити!"
Не готовий!	(:02:)	Відповідь на запит про готовність контролера до прийому програми розпуску. Формується у випадку, якщо пристрій працює в ручному режимі (віджата кнопка "Авт/Руч" на пульті оператора чи рукоятка перемикача режимів встановлена в

		положення, відмінного від "Програмний". Вимагає відповіді "Зрозумів" чи "Не зрозумів, повторити!"
Перехід у ручний режим	(:11:)	Пристрій перейшов до роботи в ручному режимі (віджата кнопка "Авт/Руч" на пульті оператора при встановленій у положення "Програмний" рукоятки перемикача режиму, чи рукоятка перемикача встановлена в положення, відмінного від "Програмний" при натиснутому положенні кнопки "Авт/руч"). Вимагає відповіді "Зрозумів" чи "Не зрозумів, повторити!"
Перехід в автоматичний режим	(:02:)	Пристрій перейшов до роботи в автоматичному режимі (натиснута кнопка "Авт/Руч" на пульті оператора при встановленій у положення "Програмний" рукоятки перемикача режиму, чи рукоятка перемикача встановлена в положення "Програмний" при натиснутому положенні кнопки "Авт/руч".

		Вимагає відповіді "Зрозумів" чи "Не зрозумів, повторити!"
Натиснуто кнопку "Скидання"	(:16:)	Фіксується факт натискання кнопки "Скидання". Вимагає відповіді "Зрозумів" чи "Не зрозумів, повторити!"
Натиснуто кнопку "Просування"	(:17:)	Фіксується факт натискання кнопки "Просування". Вимагає відповіді "Зрозумів" чи "Не зрозумів, повторити!"
Натиснуто кнопку "Затримка"	(:18:)	Фіксується факт натискання кнопки "Затримка". Вимагає відповіді "Зрозумів" чи "Не зрозумів, повторити!"
Зрозумів	(:10:)	Відповідь на ряд повідомлень АРМа у випадку їхнього правильного формату при верифікації.
Не зрозумів, повторити!	(:09:)	Відповідь на ряд повідомлень АРМа у випадку їхнього неправильного формату при верифікації.

3.2.3 Структура повідомлень обміну інформацією між АРМом і АСК ВП УЗ

Обмін повідомленнями між АРМом і АСК ВП УЗ полягає в одержанні даних про прибуття поїзда в парк прибуття (повідомлення 201) і сортувального листа для поїзда, що знаходиться на коліях парку прибуття і готового до розпуску. Ці дані АРМ одержує автоматично по мірі їхньої готовності. При цьому використовується стандартна програма розсилання повідомлень по абонентах,

використовувана на сортувальних станціях (програма «Многопротокольный маршрутизатор»). Дана програма автоматично забезпечує отримання повідомлення від АСК ВП УЗ у вигляді файлів. Ці файли розміщуються у спеціальній директорії, визначеній у настройках програми «Многопротокольный маршрутизатор». АРМ через визначені інтервали часу переглядає вміст даної директорії і у випадку знаходження нових файлів проводить їх обробку та видачу нової інформації на екран.

Формат 201 повідомлення наступний:

- код повідомлення (3 знаки);
- ЕСР станції формування (6 знаків);
- номер поїзда (4 знаки);
- станція формування поїзда (4 знаки);
- порядковий номер поїзда (3 знаки);
- станція призначення (4 знаки);
- напрямок прибуття (5 знаків);
- число (2 знаки);
- місяць (2 знаки);
- година (2 знаки);
- хвилини (2 знаки);
- номер парку (2 знаки);
- номер колії (2 знаки);
- код роботи з локомотивом (1 знак).

Повідомлення починається зі знака (: і закінчується знаком :). Між полями повідомлення ставитися пробіл.

Приклад такого повідомлення:

«(:201 150003 2268 1980 003 8500 31517 01 12 09 27 02 01 0:)».

Одержавши дане повідомлення в АРМі автоматично відображається про заняття шляху парку прибуття з відповідними атрибутами прибулого поїзда.

Після формування сортувального листа дані про процес сортування надходять в АРМ.

Сортувальний лист містить наступні дані:

1. Шапка повідомлення:

– заголовок;
– номер поїзда;
– індекс поїзда;
– час;
– номер парку;
– номер колії;
– кількість вагонів;
– умовна довжина поїзда;
– вага поїзда;
– номер головного вагона.

2. Інформаційна частина повідомлення.

– номер хвостового вагона;
– номер відчепа;
– номер колії розпуску відчепа;
– кількість фізичних вагонів у відчепі;
– вага відчепа;
– умовні оцінки;
– оцінка про ролики;
– вагова категорія.

Приклад сортувального листа наступний:

ОЦ ПРІДН-71 01.12 09-00

Сортувальний лист

розпуску з сторони основної гірки

у сортувальний парк 01

3552 4503 22 4500 07-45 02/02

23 ваг. 39.06 под. 937 т.

91831511

01 07	1	56 8 У	91831511	
02 08	1	46 8 У	91854216	
03 11	2	111 8 У	91818443	
04 14	1	37 1 О	67824185	
05 05	1	42 1 УО	60659273	
06 07	1	0 1	32237976	ПОР
07 11	1	0 1	20251781	ПОР
08 18	1	0 1	24429870	ПОР
09 18	2	58 8 О	94814159	ЛВ
10 14	1	32 8 О	94550654	
11 13	3	0 1	67611905	ПОР
12 18	15	409 8 О	94790250	ЛВ

5/1 7/2 8/1 11/3 13/3 14/2 18/18

4 ПРОГРАМНО-АПАРАТНА РЕАЛІЗАЦІЯ АРМУ ОПЕРАТОРА

4.1 Розробка інтерфейсної частини проекту

За допомогою конструктора форми, який є частиною системи програмування Delphy 7, мною були створені вісім форм, які виконують різні функції програми, рис. 4.1-4.7.

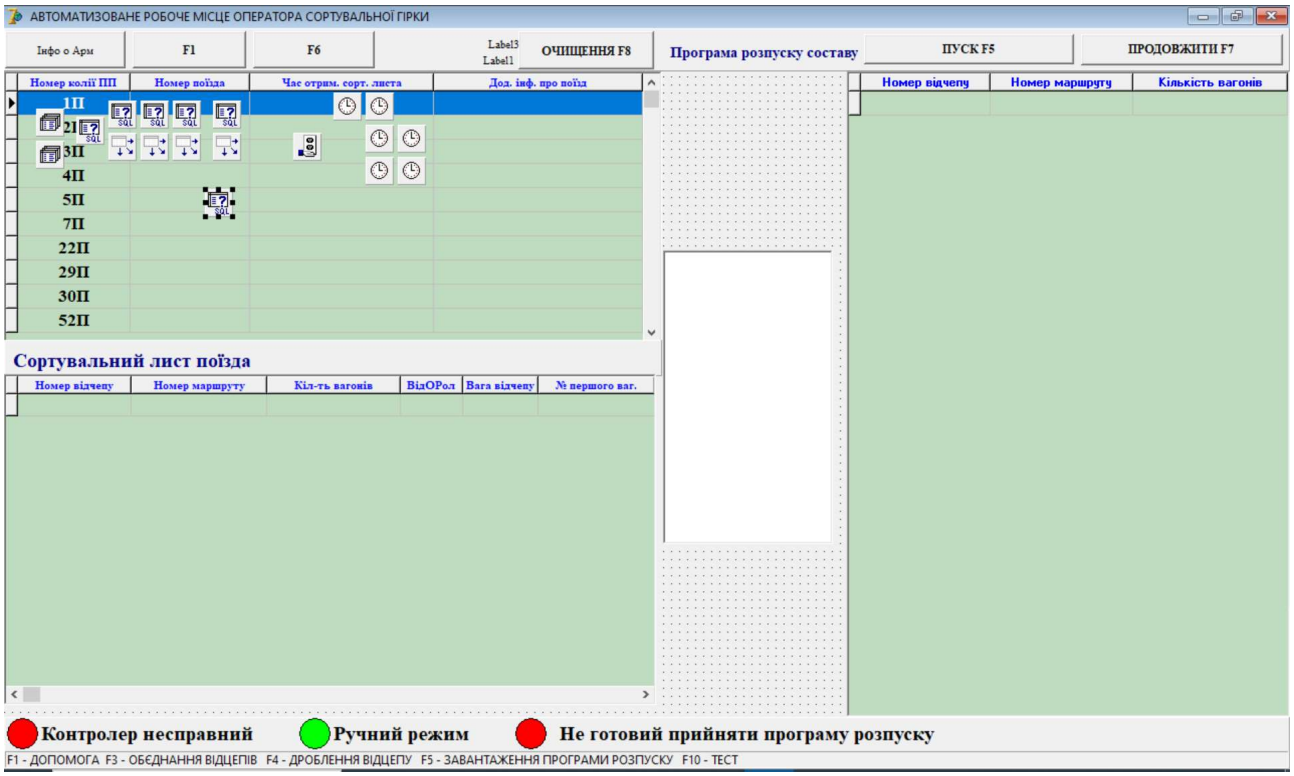

Рисунок 4.1 – Вікно головної форми(main.dfm)

Головна форма відображає оператору наступну інформацію:

– інформація про поїзди в парках прибуття (у вигляді таблиці);

– сортувальний лист (у вигляді таблиці);

– програма розпуску (у вигляді таблиці);

– стан контролера (у вигляді індикатора);

– режим розпуску поїзда (у вигляді індикатора);

– готовність контролера до прийому програми розпуску (у вигляді індикатора).

Дана інформація змінюється автоматично при виникненні відповідних подій.

Безпосередньо оператором виконуються наступні дії:

– редагування програми розпуску (об’єднання та дроблення відчепів);

– відправлення в контролер програми розпуску;

– друк сортувального листа та програми розпуску.

Вищеперераховані дії виконуються шляхом натиснення відповідної кнопки на формі або клавіатурі.

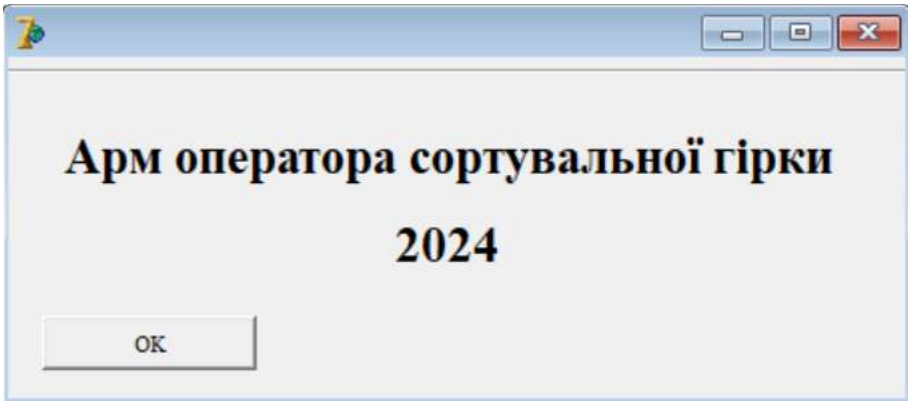

Рисунок 4.2 – Вікно інфо форми(About.dfm) - форма має цнформаційний характер

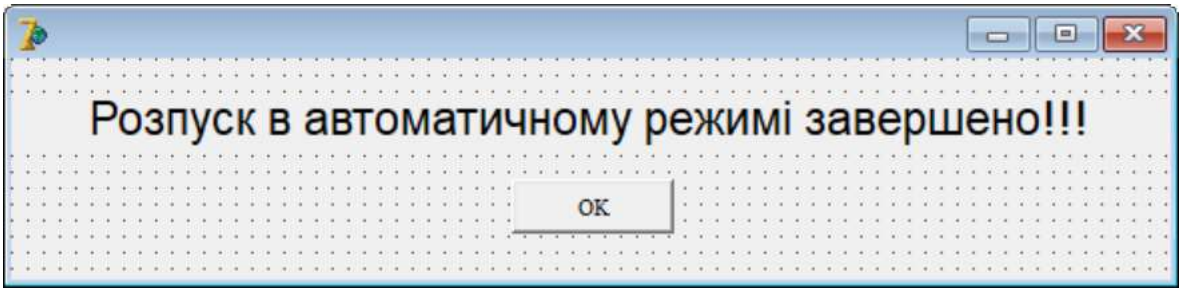

Рисунок 4.3 – Вікно форми повідомлення про розпуск ставу(MyShowMess.dfm) –має інформаційний заракетр

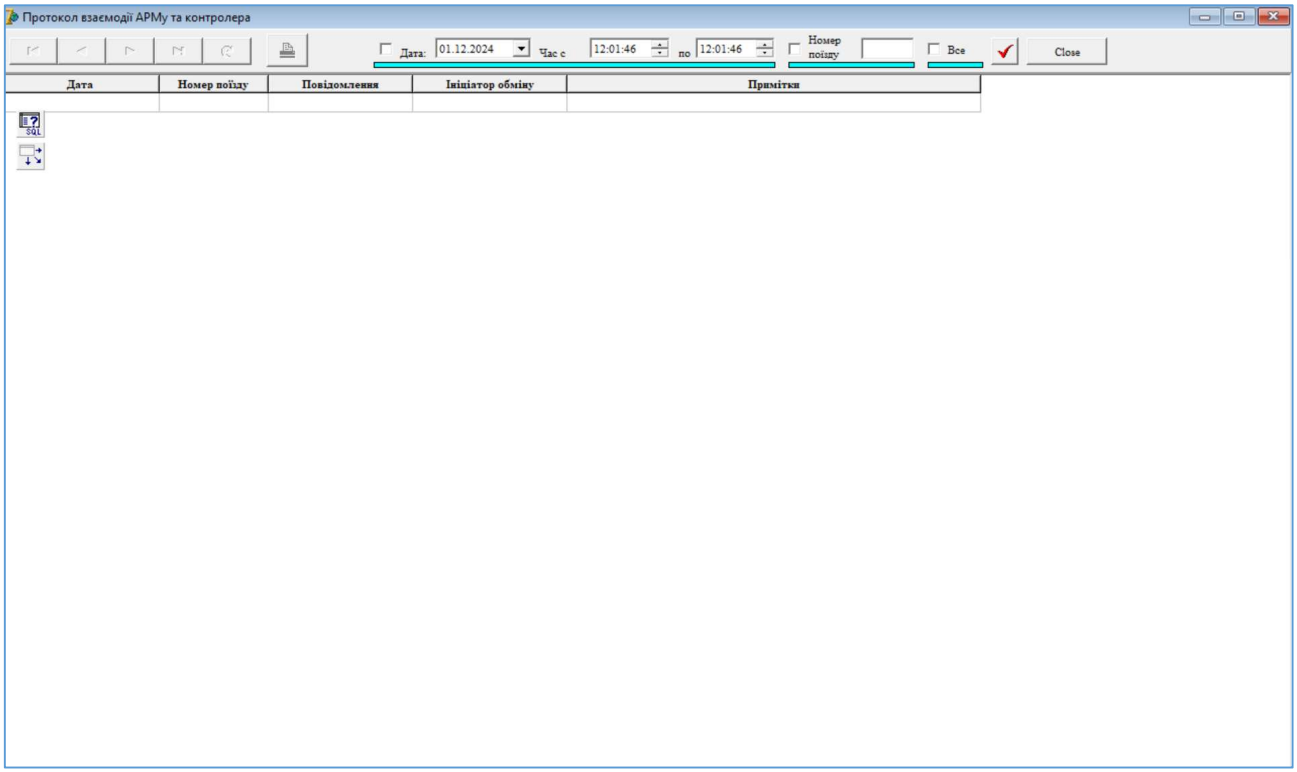

Рисунок 4.4 – Вікно форми протокол взаємодії АРМу та контролера (Protocol.dfm) – відображає перелік повідомлень між ними

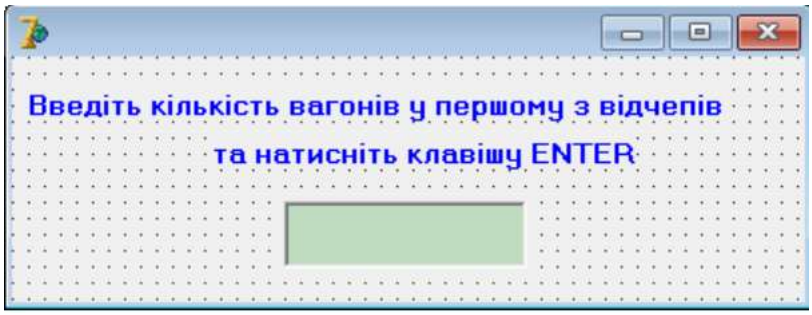

Рисунок 4.5 – Вікно форми дроблення відцепу (QuanCar.dfm)

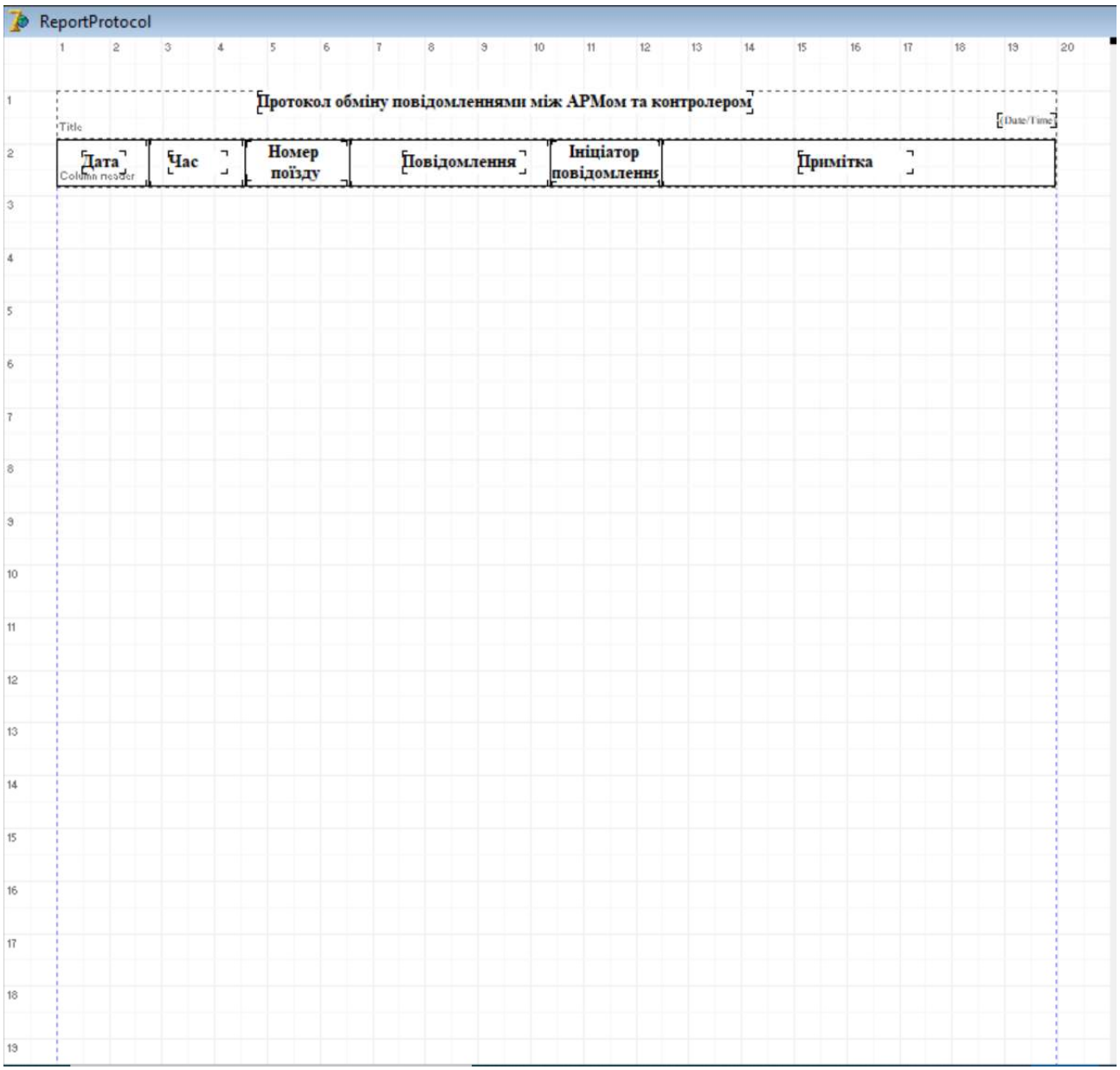

Рисунок 4.6 – Вікно форми друку протокола взаємодії АРМу та контролера (ReportProtocol.dfm) – друк на принтер протокола

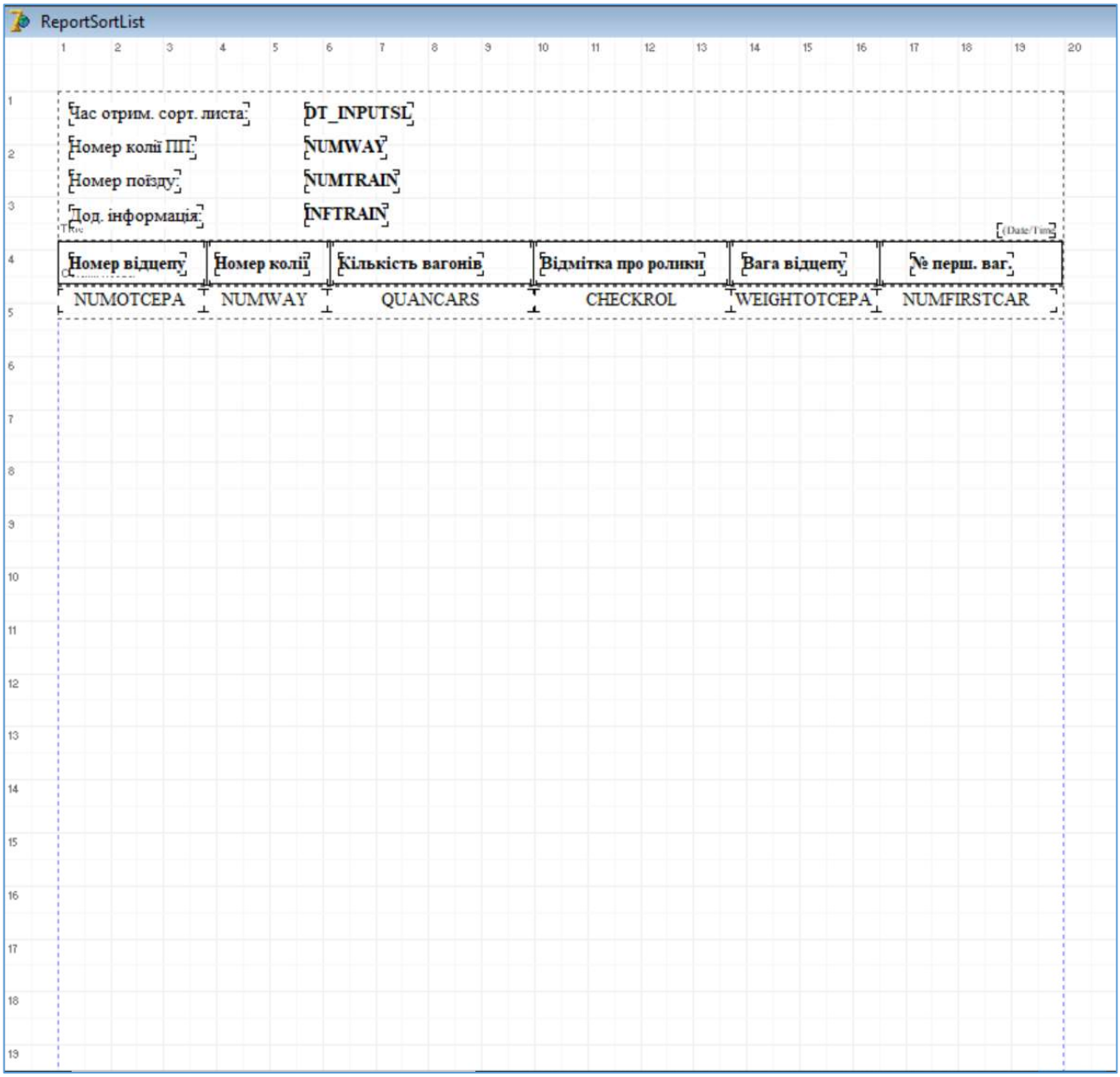

Рисунок 4.7 – Вікно форми друку сортувального листа(ReportSortList.dfm) – друк на принтер сортувального листа

4.2. Розробка програмного кода проекту

На другому етапі розробки програми я наповнив компоненти програмним кодом, призначеним для розв’язання поставлених задач. У результаті маю вісім блоків програмного коду:

- Main.pas;
- About.pas;
- Help.pas;
- MyShowMess.pas;

- Protocol.pas;
- CuanCar.pas;
- RepProtocol.pas;
- RepSortList.pas.

Програмний код блоків представлений у ДОДАТКУ А.

4.3 Компіляція та запуск виконуєго файлу

Після компіляції та виправлення помилок, що виникли, отримав виконуваний файл «Арм.exe», для застосування в операційній системі Windows.

Запускаєм файл на ПК (з рекомендованими параметрами) в операційній системі Windows 10.

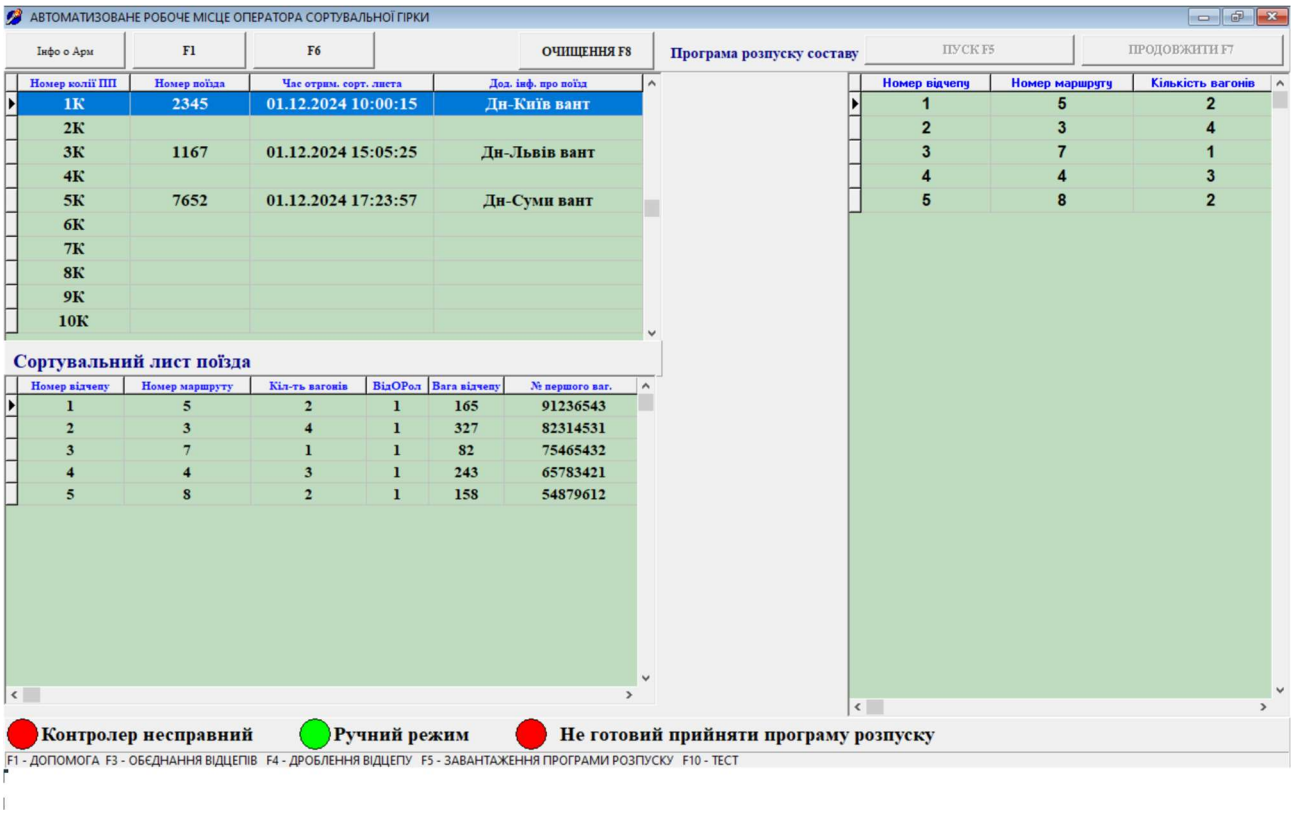

Рисунок 4.8 – головний екран АРМа - в ПП знаходиться состав №2345, на екрані відображається сортувальний лист і програма розпуску для нього

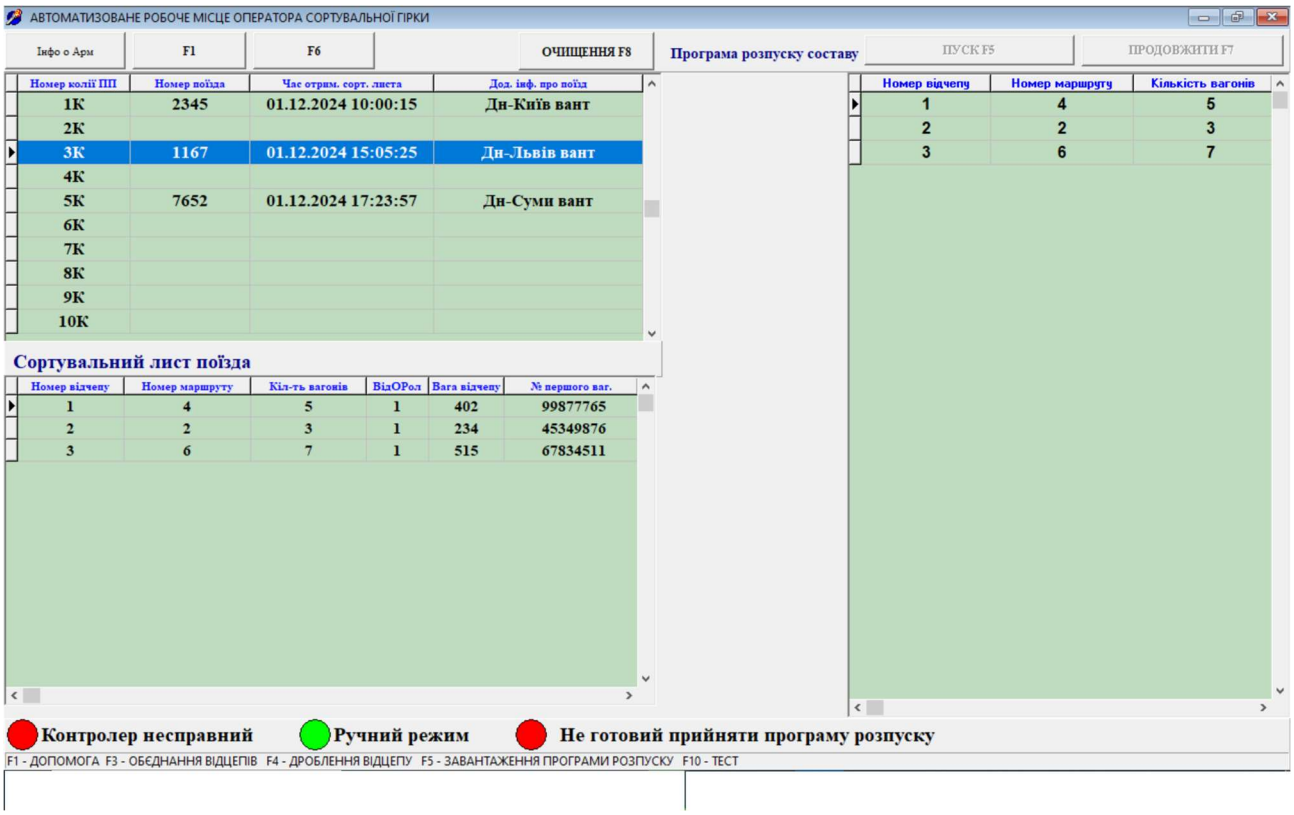

Рисунок 4.9 – голов. екран АРМа - в ПП знаходиться состав №1167, на екрані відображається сортувальний лист і програма розпуску для нього

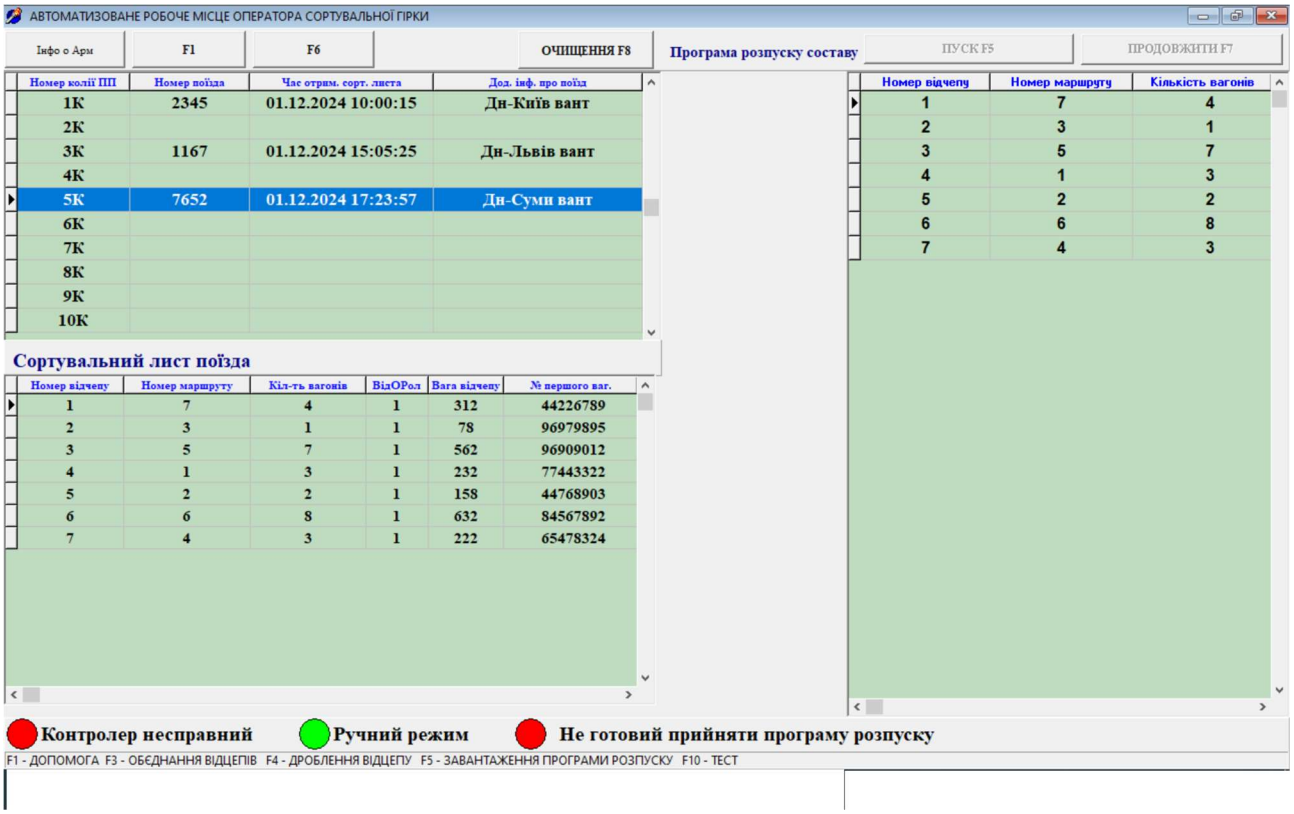

Рисунок 4.10 – головний екран АРМа - в ПП знаходиться состав №7652, на екрані відображається сортувальний лист і програма розпуску для нього

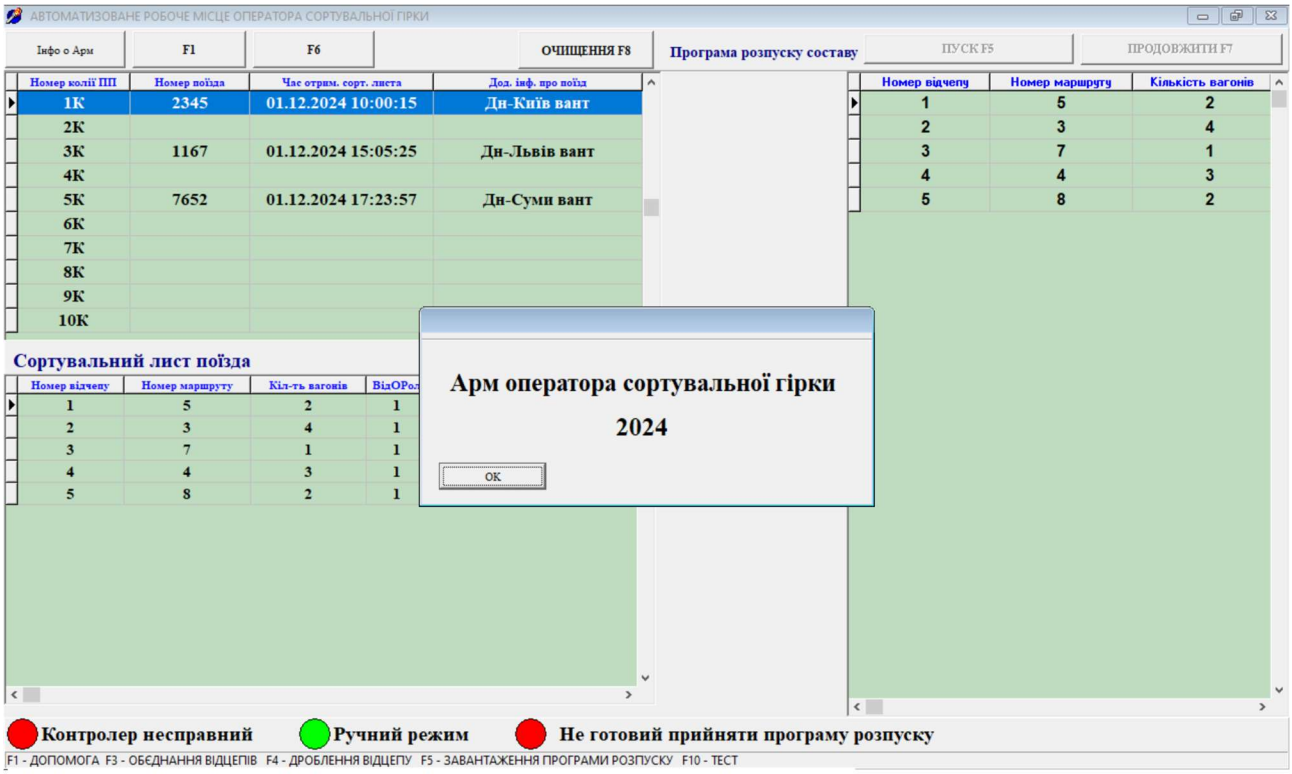

Рисунок 4.11 – при натисканні кнопки «Інфо о Арм» - маємо інформаційне повідомлення

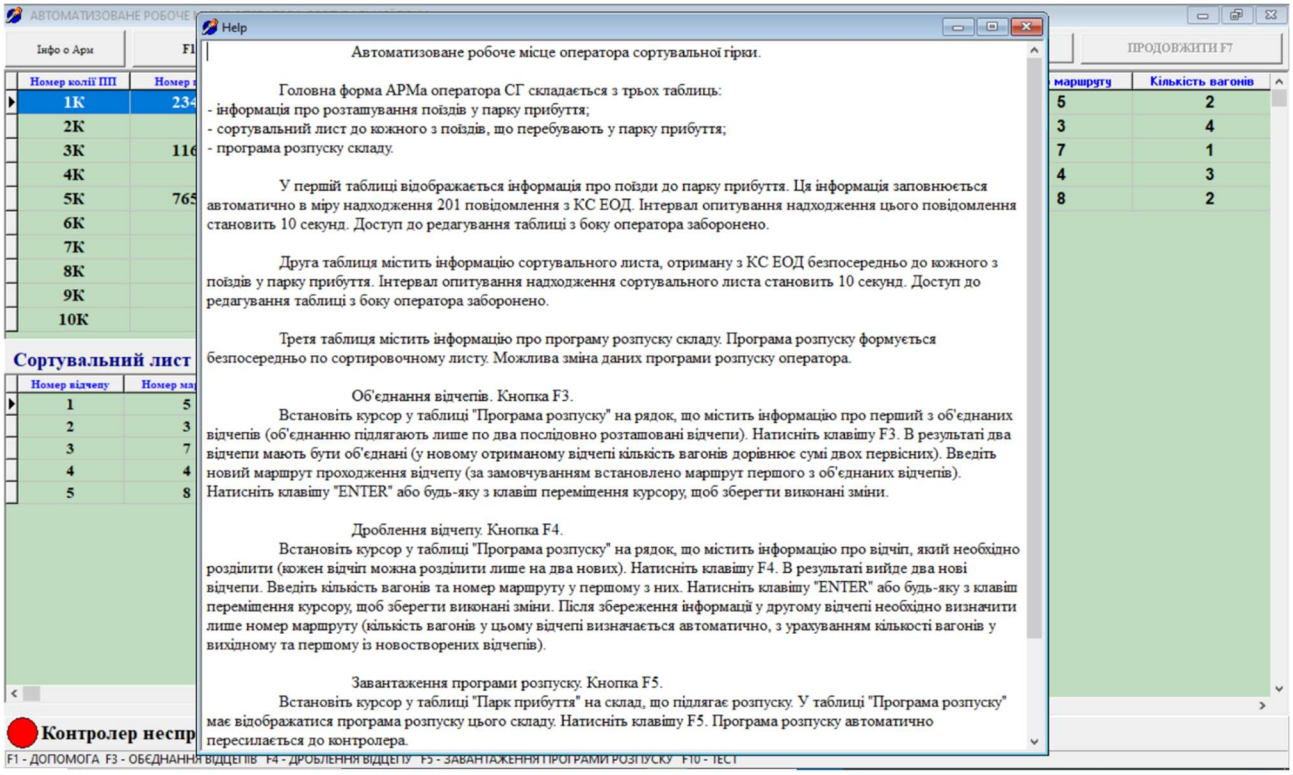

Рисунок 4.12 – при натисканні кнопки «F1» - маємо вікно яке містить довідкову інформацію

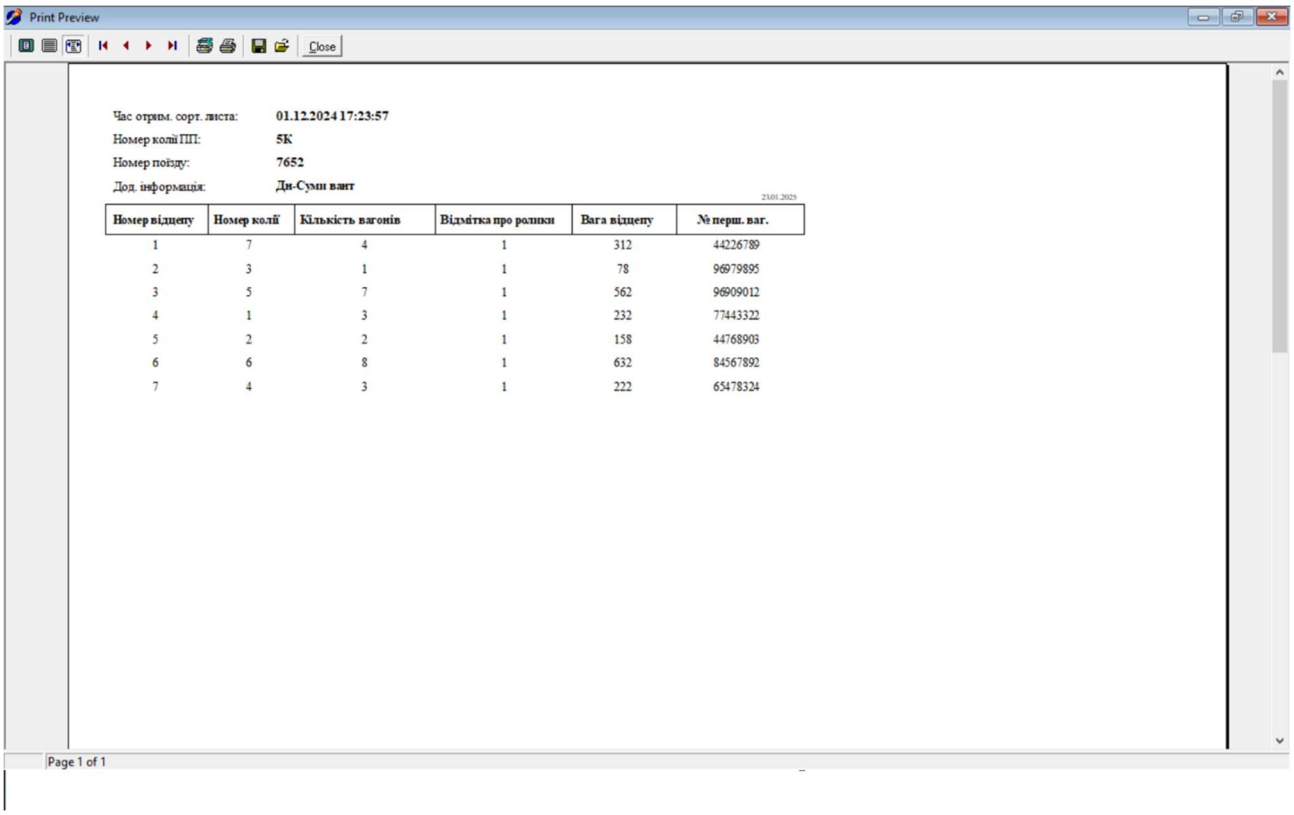

Рисунок 4.13 – при натисканні кнопки «F6» - маємо вікно форми друка сортувального листа

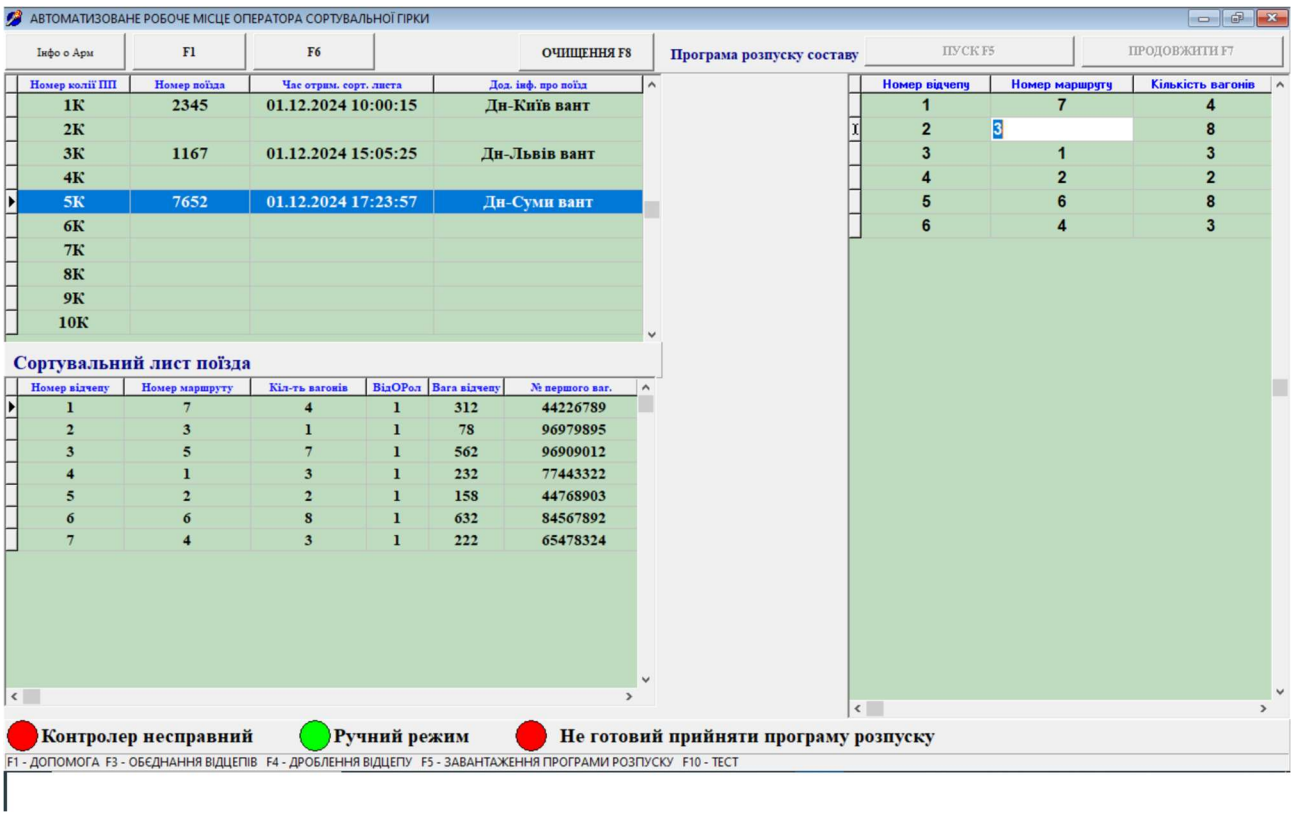

Рисунок 4.14 – при натисканні кнопки «F3» - маємо результат об’єднання двох відчепів і вибір маршруту для них

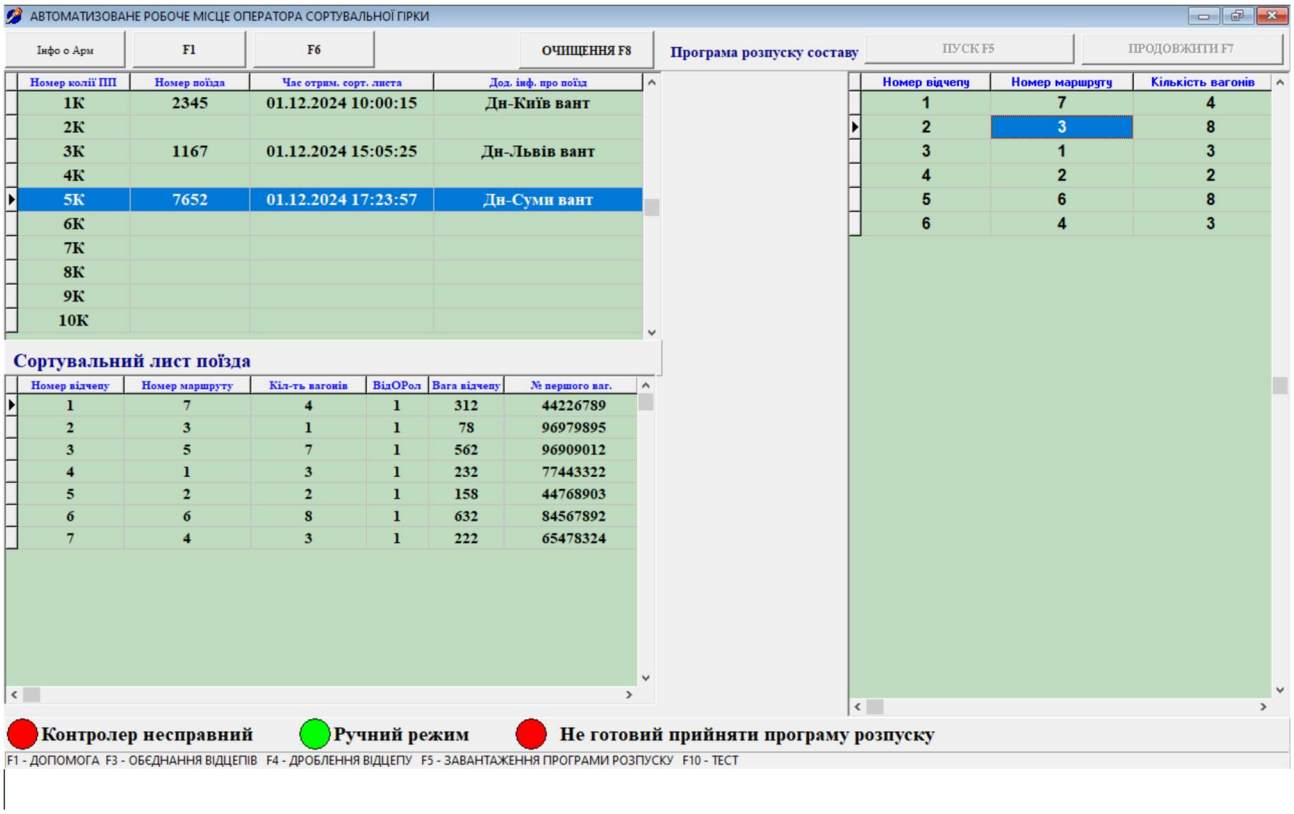

Рисунок 4.15 – при натисканні кнопки «F3» - маємо результат об’єднання двох відчепів і підтвердження маршруту для них

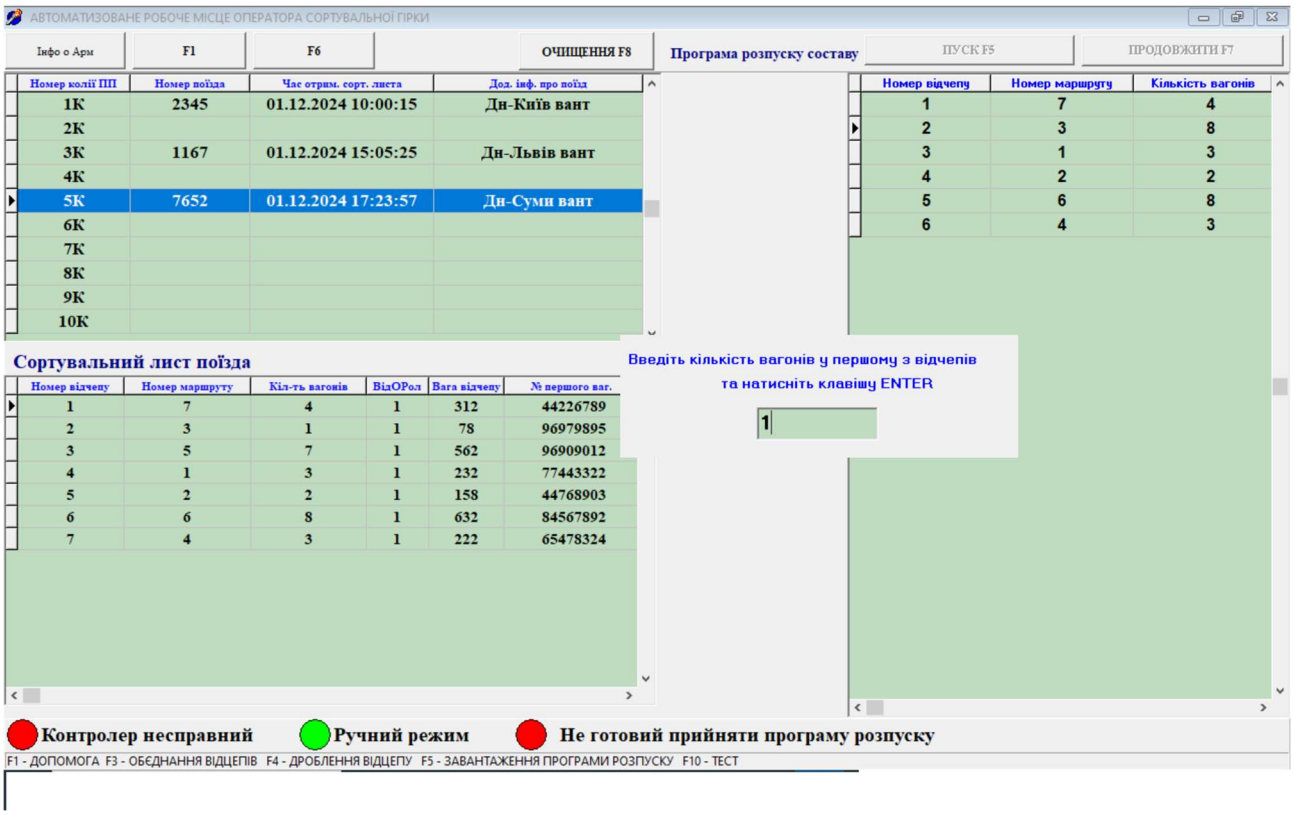

Рисунок 4.16 – при натисканні кнопки «F4» - дроблення двох відчепів

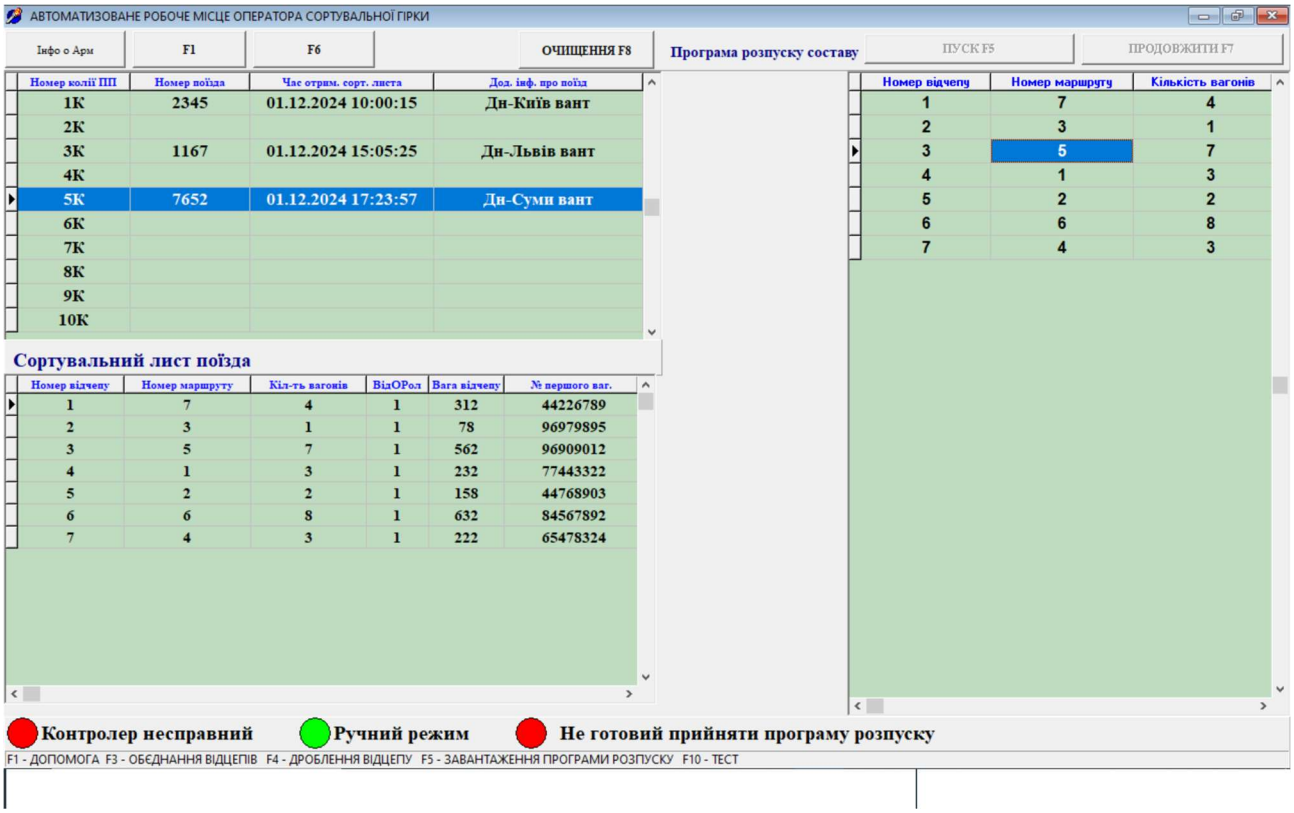

Рисунок 4.17 – при натисканні кнопки «F4» - результат дроблення двох відчепів

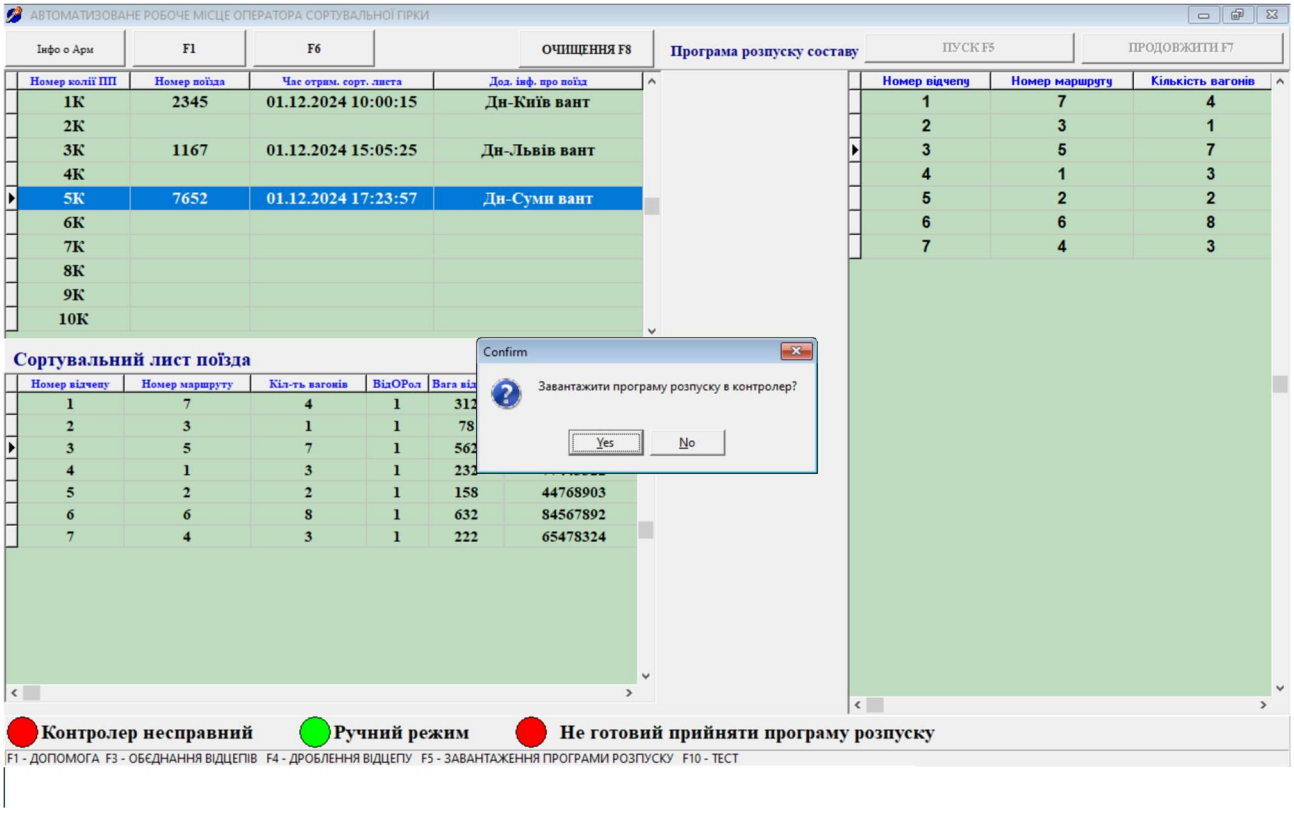

Рисунок 4.18 – при натисканні кнопки «F5» - відправлення програми розпуску в контролер

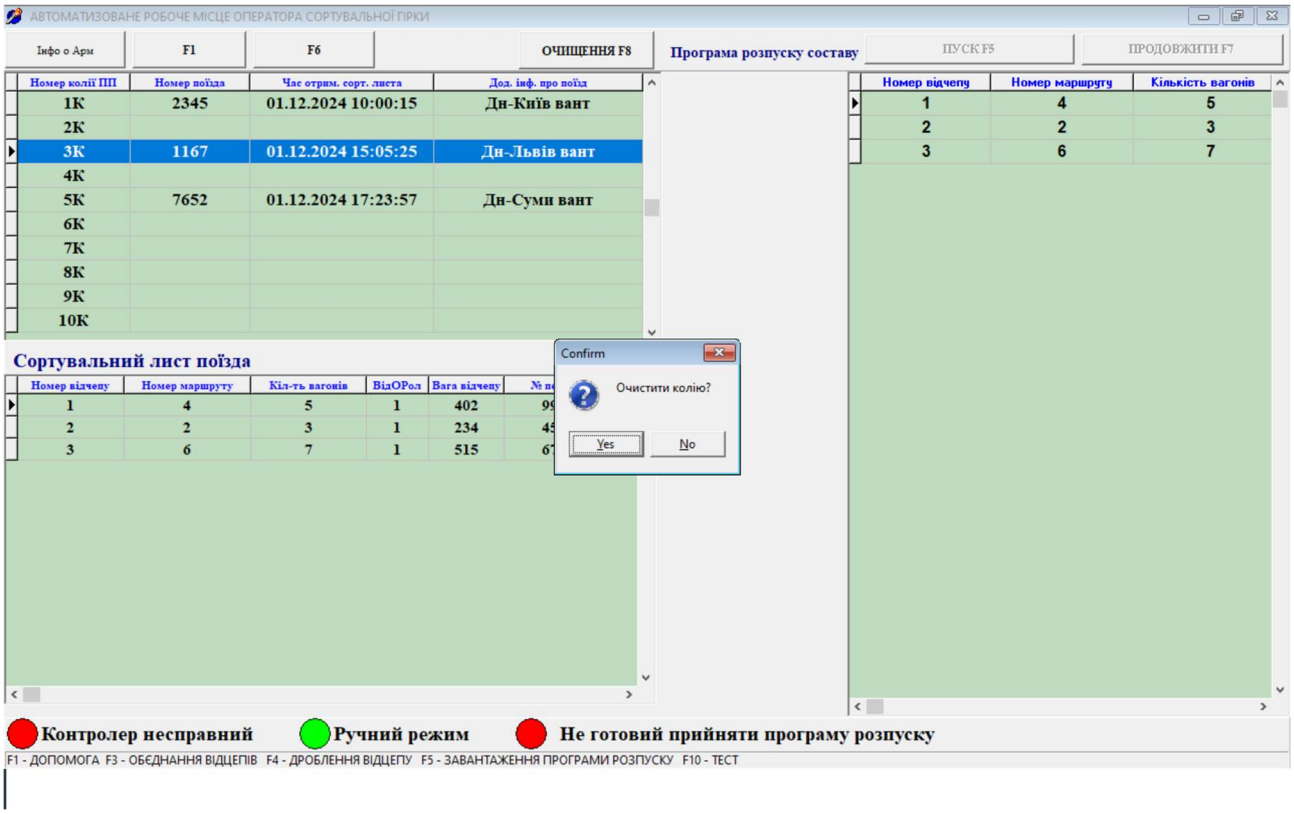

Рисунок 4.19 – при натисканні кнопки «F8» - запрос на очистку колії

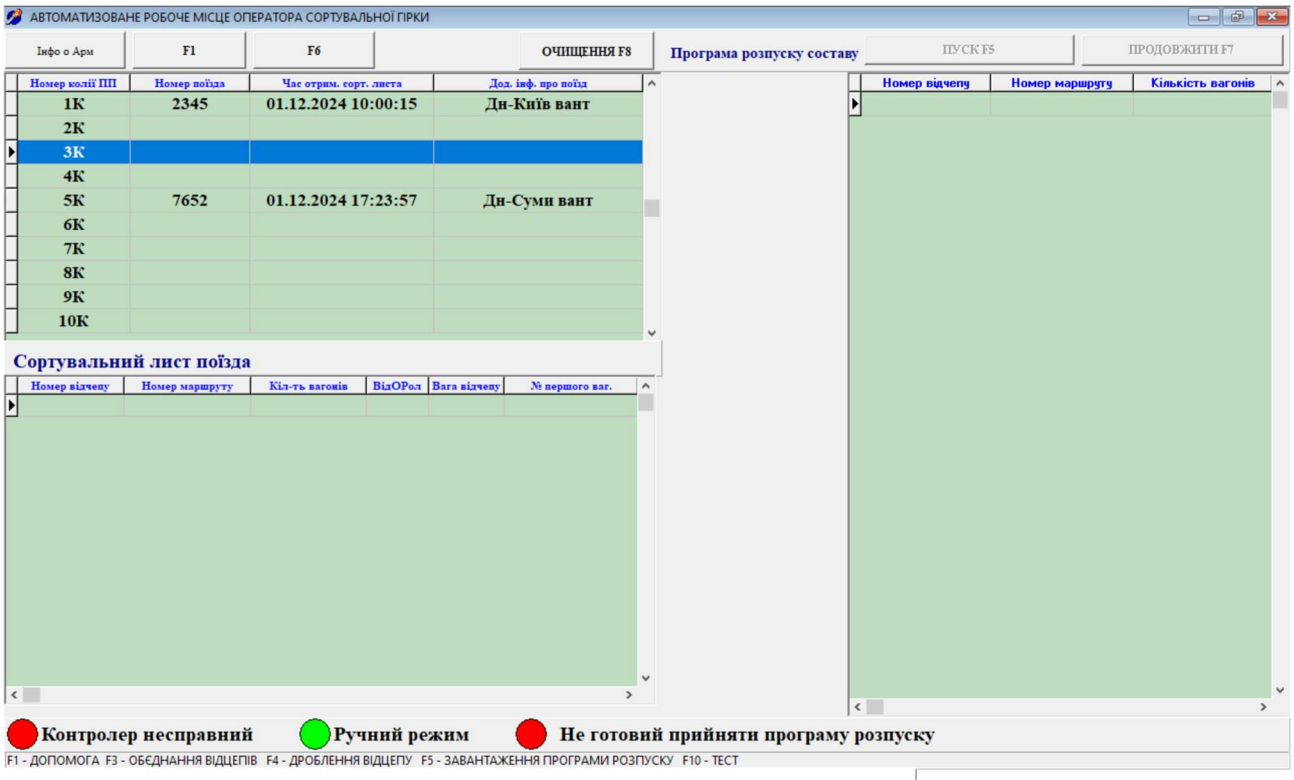

Рисунок 4.20 – при натисканні кнопки «F8» - та підтвердження запросу на очистку колії

Перевірив коректну роботу основних функцій програми в імітаційних умовах.

ВИСНОВКИ ТА РЕКОМЕНДАЦІЇ

Одним із основних механізмів підвищення ефективності операційної діяльності сортувальної станції є автоматизація технологічного процесу.

Автоматизація – це вирішальна ланка технологічної роботи сортувальної станції.

За умов відсутності значних капітальних вкладень у розвиток інфраструктури багато проблем можна вирішити, створивши якісну систему керування сортувальної станції, що дозволить підвищити безпеку руху, пропускну спроможність, прискорити швидкість розпуску составів, покращити ефективність техгологічного процесу.

Автоматизація технологічних процесів – це етап комплексної автоматизації, що характеризується звільненням людини від безпосереднього виконання функцій управління технологічними процесами і передачею цих функцій автоматичним пристроям.

При автоматизації технологічних процесів отримання, перетворення, передача і використання енергії, матеріалів і інформації виконуються автоматично за допомогою спеціальних технічних засобів та систем управління.

Аналіз останніх досліджень і публікацій. Завдання, пов’язані з розробкою, модернізацією та впровадженням сучасних систем автоматизації, свідчать про актуальність пошуку відповідей на проблемні питання. У низці досліджень розглянуто питання впровадження сучасних інформаційних технологій , зокрема автоматизованої системи керування вантажними перевезеннями (далі – АСК ВП УЗ-Є), яка ґрунтується на електронному документообігу.

ПЕРЕЛІК ПОСИЛАНЬ

1. Бульба С.С., Лукова-Чуйко Н.В., Лелет І.В. Система виконання сервісів Укрзалізниці як композитних додатків у розподіленій мережі. Інформаційно-керуючі системи на залізничному транспорті. 2018. № 2. С. 38–42.

2. Вернигора Р.В., Єльнікова Л.О. Структура та принципи функціонування прогнозної моделі роботи залізничного напрямку. Транспортні системи та технології перевезень. 2015. Вип. 9. С. 16–22.

3. Чернецька-Білецька Н.Б., Павлюченко В.О., Кононенко С.В. Аналіз систем автоматизації управління технологічними процесами на станціях залізничного транспорту. Вісник Інженерної академії України. 2013. Вип. 3–4. С. 185–187.

4. Лаврухін О.В. Формування підходів щодо реалізації системи підтримки прийняття рішень опера- тивного управління поїздопотоками з розподіленим штучним інтелектом. Транспортні системи та тех- нології перевезень. 2014. Вип. 8. С. 88–99.

5. Bardas, O., Skovron I., Demchenko Y. and others. Influence research of traffic prediction accuracy on effective management of the trains breaking-up order. Transport Problems. International scientific journal. 2017, Volume 12, Issue 1. Gliwice, 2017. P. 151–158.

6. Бардась О.О. Удосконалення інтелектуальних технологій виконання поїзної роботи на сортувальних станціях. Транспортні системи та технології перевезень. 2016. Вип. 11. С. 9–15.

7. Bux M., Leser U. Parallelization in Scientific Workflow Management Systems. Distributed, Parallel, and Cluster Computing. 2013. № 1. P. 24.

8. Chandrappa S., Dharmanna L., Shubhada V.P., Meghana N.U. Automatic Control of Railway Gates and Destination Notification System using Internet of Things (IoT). International Journal of Education and Management Engineering. 2017. № 7 (5). P. 45–55.

9. Kyrychenko H., Statyvka Y., Strelko O., Berdnychenko Y., Nesterenko H. Assessment of cargo delivery quality using fuzzy set apparatus. International Journal of Engineering & Technology. 2018. № 7 (4.3). P. 262–265.

10. Strelko O., Kyrychenko H., Berdnychenko Y., Hurinchuk S. Automation of Work Processes at Ukrainian Sorting Stations. International Journal of Engineering & Technology. 2018. № 7 (2.23). P. 516–518.
11. Жуковицкий И.В., Егоров О.И. Процедура идентификации поездов с использованием информации АСК ВП УЗ Е. Інформаційно-керуючі системи на залізничному транспорті. 2015. № 6 (115) С. 61–66.
12. Грицунов О.В. Інформаційні системи та технології : навчальний посібник. Харків : ХНАМГ, 2010. 222 с.
13. Ronny S. Hansmann and Uwe T. Zimmermann. Optimal sorting of rolling stock at hump yards. In Hans-Joachim Krebs and Willi J¨ager, editors, Mathematics, Key Technology for the Future, pages 189–203. Springer Berlin Heidelberg, 2008.
14. K. Kube. Progressive Railroading, 2002. №7, с. 50–52.

ДОДАТОК А

А.1 Лістінг Main.pas

```
unit Main;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Grids, DBGrids, ExtCtrls, DBTables, StdCtrls, Buttons, Db, ComCtrls,
  FileCtrl, ScktComp;

type
  TfrMain = class(TForm)
    Panel1: TPanel;
    dbMain: TDatabase;
    Panel2: TPanel;
    shJobKontr: TShape;
    shMode: TShape;
    shReady: TShape;
    lContr: TLabel;
    lMode: TLabel;
    lReady: TLabel;
    gTrainPP: TDBGrid;
    Panel3: TPanel;
    Label4: TLabel;
    Panel4: TPanel;
    Panel5: TPanel;
    qTrainPP: TQuery;
    qSortList: TQuery;
    qProgrRospusk: TQuery;
    dsTrainPP: TDataSource;
    dsSortList: TDataSource;
    dsProgrRospusk: TDataSource;
    gSortList: TDBGrid;
    qSortListNUMOTCEPA: TIntegerField;
    qSortListNUMWAY: TStringField;
    qSortListQUANCARS: TIntegerField;
    qSortListCHECKROL: TStringField;
    qSortListWEIGHTOTCEPA: TStringField;
    qSortListNUMFIRSTCAR: TStringField;
    gProgrRospusk: TDBGrid;
    qProgrRospuskPROGRROSPUSK_ID: TIntegerField;
    qProgrRospuskTRAINPP_ID: TIntegerField;
    qProgrRospuskNUMOTCEPA: TIntegerField;
    qProgrRospuskNUMWAY: TStringField;
    qProgrRospuskQUANCARS: TIntegerField;
    qProgrRospuskCHECKROL: TStringField;
    qSortListSORTLIST_ID: TIntegerField;
```

```
qSortListTRAINPP_ID: TIntegerField;
sbF6: TSpeedButton;
sbF1: TSpeedButton;
SpeedButton6: TSpeedButton;
qAccept: TQuery;
tmSortList: TTimer;
StatusBar1: TStatusBar;
flbMessKSEOD: TFileListBox;
Label2: TLabel;
sbStart: TButton;
qTrainPPTRAINPP_ID: TIntegerField;
qTrainPPDT_INPUTSL: TDateTimeField;
qTrainPPNUMTRAIN: TStringField;
qTrainPPINFTRAIN: TStringField;
qTrainPPPROGRROSP_YES: TStringField;
tmContrReady: TTimer;
qStoreData: TQuery;
dsStoreData: TDataSource;
mProgrRospusk: TMemo;
tmWait: TTimer;
dbStore: TDatabase;
qStoreDataSTOREDATA_ID: TIntegerField;
qStoreDataDATEOPER: TDateTimeField;
qStoreDataTIMEOPER: TDateTimeField;
qStoreDataNUMTRAIN: TStringField;
qStoreDataCODEOPER: TStringField;
qStoreDataWHOSEND: TStringField;
qStoreDataREMAR: TStringField;
ServerSocket1: TServerSocket;
Timer1: TTimer;
TimerLive: TTimer;
qTrainPPNUMWAY: TStringField;
TimWaitConn: TTimer;
Label1: TLabel;
qDelData: TQuery;
Label3: TLabel;
sbClear: TSpeedButton;
sbContine: TButton;
procedure FormCreate(Sender: TObject);
procedure qSortListAfterInsert(DataSet: TDataSet);
procedure qSortListAfterOpen(DataSet: TDataSet);
procedure qProgrRospuskAfterInsert(DataSet: TDataSet);
procedure qProgrRospuskAfterOpen(DataSet: TDataSet);
procedure qProgrRospuskAfterPost(DataSet: TDataSet);
procedure qProgrRospuskBeforePost(DataSet: TDataSet);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure tmSortListTimer(Sender: TObject);
procedure FormActivate(Sender: TObject);
procedure sbStartClick(Sender: TObject);
procedure gProgrRospuskKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure SpeedButton6Click(Sender: TObject);
```

```
    procedure sbF6Click(Sender: TObject);
    procedure tmContrReadyTimer(Sender: TObject);
    procedure tmWaitTimer(Sender: TObject);
    procedure dsTrainPPDataChange(Sender: TObject; Field: TField);
    procedure sbF1Click(Sender: TObject);
    procedure qProgrRospuskAfterEdit(DataSet: TDataSet);
    procedure ServerSocket1ClientRead(Sender: TObject;
      Socket: TCustomWinSocket);
    procedure Timer1Timer(Sender: TObject);
    procedure TimerLiveTimer(Sender: TObject);
    procedure TimWaitConnTimer(Sender: TObject);
    procedure sbClearClick(Sender: TObject);
    procedure sbContineClick(Sender: TObject);
    procedure dsProgrRospuskUpdateData(Sender: TObject);
  private
    { Private declarations }
  public
  flag_ins:boolean;
    { Public declarations }
  end;

var
  frMain: TfrMain;
  CodeSortList, CodeProgrRospusk: Integer;
  PrRospuskBe, SendPrRospusk, I_OK, ContrError, ContrReady, Flag_Block,
  errCShow:boolean;
  BlockInData: Array[1..6] of Char;
  RospuskTrain:integer;
  iq, OldQuanCars, NumOtc, QuanOtv:Integer;
  CO,WS,R:String;
  MesIn:String;
  MesSend, RospuskTrainNum: String;
  SockContr: TCustomWinSocket;
  WaitConnFlag: boolean;
procedure StoreData(CO, WS, R:String);
procedure SendKontr(var StrSend:String);
procedure delDataProgrRosp;

implementation

uses About, RepSortList, Protocol, Help, MyShowMess, QuanCar;

{$R *.DFM}


procedure StoreData(CO, WS, R:String);
Begin
  with frMain do
  Begin
  qStoreData.Insert;
  qStoreData.FieldByName('DateOper').AsDateTime:=Now;
  qStoreData.FieldByName('TimeOper').AsDateTime:=Now;
  if RospuskTrain<>0 then
```

```
    qStoreData.FieldByName('NumTrain').AsString:=RospuskTrainNum
   else
    qStoreData.FieldByName('NumTrain').AsString:=qTrainPPNumTrain.Value;
   qStoreData.FieldByName('CodeOper').AsString:=CO;
   qStoreData.FieldByName('WhoSend').AsString:=WS;
   qStoreData.FieldByName('Remar').AsString:=R;
   qStoreData.Post;
   end;
end;

procedure delDataProgrRosp;
Begin
  with frMain do
    Begin
    qDelData.SQL.Clear;
    qDelData.SQL.Add('DELETE FROM SortList WHERE TrainPP_Id=:ID');
    qDelData.Params[0].AsInteger:=RospuskTrain;
    qDelData.ExecSQL;
    qDelData.SQL.Clear;
    qDelData.SQL.Add('DELETE FROM ProgrRospusk WHERE TrainPP_Id=:ID');
    qDelData.Params[0].AsInteger:=RospuskTrain;
    qDelData.ExecSQL;
    with qTrainPP do
      if locate('TrainPP_Id', RospuskTrain, [loCaseInsensitive]) then
        Begin
        qTrainPP.Edit;
        qTrainPP.FieldByName('DT_INPUTSL').AsString:='';
        qTrainPP.FieldByName('PROGRROSP_YES').AsString:='0';
        qTrainPP.FieldByName('InfTrain').AsString:='';
        qTrainPP.FieldByName('NumTrain').AsString:='';
        qTrainPP.Post;
        End;
    RospuskTrain:=0;
    RospuskTrainNum:='';
    End;
end;


procedure TfrMain.FormCreate(Sender: TObject);
begin
{ Flag_Dobl:=false;
 Flag_Block:=True; {Дозвіл прийому повідомлень через COM порт}
 {Настройка COM-порта}
{ LinkCompTwo.ComNumber:=1;  //№ порта: 1 - перший
 LinkCompTwo.Baud:=115200;    //Швидкість, бод
 LinkCompTwo.DataBits:=8;   //Кількість біт даних: 8
 LinkCompTwo.StopBits:=2;   //Кількість стопових біт: 1
 LinkCompTwo.Parity:=pNone; //pEven;//pOdd; //pNone;   //Перевірка на парність: так
}
  try
    ServerSocket1.Port:=5;
    ServerSocket1.Active:=True;
  except
```

```
   messageDlg('Помилка ініціалізації!',mtError,[mbOk],0)
 end;
 qTrainPP.Close;
 qTrainPP.Open;
 qSortList.Close;
 qSortList.Open;
 qProgrRospusk.Close;
 qProgrRospusk.Open;
 PrRospuskBe:=false;
 qStoreData.Open;
 iq:=0;
 TimerLive.Enabled:=false;
 TimerLive.Interval:=5000;
 TimerLive.Enabled:=true;
 errCShow:=false;
 flag_ins:=true;
end;

procedure TfrMain.qSortListAfterInsert(DataSet: TDataSet);
begin
  qSortList.FieldByName('SortList_Id').ASINTEGER:=0;
end;

procedure TfrMain.qSortListAfterOpen(DataSet: TDataSet);
begin
 WITH qSortList DO
   Locate('SortList_Id',
        CodeSortList,[loCaseInsensitive]);
 CodeSortList:=0;
end;

procedure TfrMain.qProgrRospuskAfterInsert(DataSet: TDataSet);
begin
  qProgrRospusk.FieldByName('ProgrRospusk_Id').ASINTEGER:=0;
end;

procedure TfrMain.qProgrRospuskAfterOpen(DataSet: TDataSet);
Var QuanCars:integer;
begin
 WITH qProgrRospusk DO
   Locate('ProgrRospusk_Id',
        CodeProgrRospusk,[loCaseInsensitive]);
 CodeProgrRospusk:=0;
{  if Flag_Dobl then
    Begin
    QuanCars:=qProgrRospuskQuanCars.Value;
    qProgrRospusk.Next;
    qProgrRospusk.Edit;
    qProgrRospusk.FieldByName('QuanCars').AsInteger:=OldQuanCars-QuanCars;
    OldQuanCars:=0;
    Flag_Dobl:=false;
    qProgrRospusk.Post;
    qProgrRospuskNumWay.FocusControl;
```

```
    qProgrRospusk.Edit;
    End;}
end;

procedure TfrMain.qProgrRospuskAfterPost(DataSet: TDataSet);
begin
  qProgrRospusk.Close;
  qProgrRospusk.Open;
end;

procedure TfrMain.qProgrRospuskBeforePost(DataSet: TDataSet);
begin
   if qProgrRospusk.State=dsInsert then
    Begin
    qAccept.SQL.Clear;
    qAccept.SQL.Add('SELECT GEN_ID(ProgrRospusk_Gen,0)');
    qAccept.SQL.Add('FROM RDB$DATABASE');
    qAccept.Active:=true;
    CodeProgrRospusk:=qAccept.FieldByName('Gen_Id').Value;
    CodeProgrRospusk:=CodeProgrRospusk+1;
    qAccept.Active:=false;
    End;
end;

procedure TfrMain.FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
var F:TextFile;
    i, n_str, m:integer;
    SMess:ARRAY[1..1000] of String;
    NumTrain, NumWay, tt:String;
begin
  Case Key of
//    VK_F3: sbF3Click(Sender);
//    VK_F4: sbF4Click(Sender);
    VK_F6:   ReportSortList.Preview;
//    VK_F6: sbF6Click(Sender);
    VK_F1: sbF1Click(Sender);
    VK_F5: sbStartClick(Sender);
    VK_F7: sbContineClick(Sender);
    VK_F8: sbClearClick(Sender);
    VK_F9: frProtocol.Show;
    VK_F10: Begin
          if MessageDlg('Завантажити тест-дані на 1-у колію?',
            mtConfirmation, [mbYes, mbNo], 0) = mrYes then
          Begin
          flbMessKSEOD.Directory:='d:\Arm\Test\';
          flbMessKSEOD.Update;
          flbMessKSEOD.ItemIndex:=0;
          if (flbMessKSEOD.Items[0]<>'[..]') and (qProgrRospusk.State<>dsInsert) and (qProgrRospusk.State<>dsEdit) then
            begin
            AssignFile(F, flbMessKSEOD.Items[0]);
            Reset(F);
            i:=0;
```

```
          while not eof(F) do
           Begin
           i:=i+1;
           readln(F, SMess[i]);
           End;
          n_str:=i;
          if (Copy(SMess[1], 1, 5)='(:201') and (Copy(SMess[1],Pos('/',SMess[1])-1,1)='2') then
           begin
           with qTrainPP do
           if (locate('NumWay', IntToStr(StrToInt(Copy(SMess[1],Pos('/',SMess[1])+1,2)))+'П', [loCaseInsensitive])) and
             (Copy(SMess[1],Pos('/',SMess[1])-1,1)='2') then
            Begin
            qTrainPP.Edit;
            qTrainPP.FieldByName('NumTrain').AsString:=Copy(SMess[1], 13, 4);
            qTrainPP.Post;
            End
          end
         else
          if  n_str>10 then
           Begin
           i:=1;
//       while (Length(SMess[i])<>8) and (i<n_str) do
           while (Copy(SMess[i], 1, 3)<>' 01') and (i<n_str) do
            i:=i+1;
           NumTrain:=Copy(SMess[i-3], 1, 4);
           tt:=Copy(SMess[i-3], Length(SMess[i-3])-1, 2);
           NumWay:=IntToStr(StrToInt(Copy(SMess[i-3], Length(SMess[i-3])-1, 2)));
           with qTrainPP do
           if locate('NumWay', NumWay+'П', [loCaseInsensitive]) then
            Begin
            if qTrainPPPROGRROSP_YES.Value='1' then
             begin
             qProgrRospusk.First;
             while not qProgrRospusk.eof do
              qProgrRospusk.delete;
              qProgrRospusk.Close;
              qProgrRospusk.Open;
              qSortList.first;
              while not qSortList.eof do
               qSortList.delete;
             qSortList.Close;
             qSortList.Open;
             locate('NumWay', NumWay+'П', [loCaseInsensitive]);
             end;
//         i:=i+1;
             while (i<n_str) and (SMess[i]<>'') and (Copy(SMess[i], 1, 1)=' ') do
              Begin
              qSortList.Insert;
              qSortList.FieldByName('TrainPP_Id').AsInteger:=qTrainPPTrainPP_Id.Value;
              qSortList.FieldByName('NUMOTCEPA').AsInteger:=StrToInt(Copy(SMess[i], 2, 2));
              qSortList.FieldByName('NUMWAY').AsString:=Copy(SMess[i], 6, 2);
              qSortList.FieldByName('QUANCARS').AsInteger:=StrToInt(Copy(SMess[i], 11, 2));
              qSortList.FieldByName('CHECKROL').AsString:=Copy(SMess[i], 20, 1);
```

```
                qSortList.FieldByName('WEIGHTOTCEPA').AsString:=Copy(SMess[i], 15, 4);
                qSortList.FieldByName('NUMFIRSTCAR').AsString:=Copy(SMess[i], 28, 8);     {!!!}
                qSortList.Post;
                flag_ins:=false;
                qProgrRospusk.Insert;
                qProgrRospusk.FieldByName('TrainPP_Id').AsInteger:=qTrainPPTrainPP_Id.Value;
                qProgrRospusk.FieldByName('NUMOTCEPA').AsInteger:=StrToInt(Copy(SMess[i], 2, 2));
                qProgrRospusk.FieldByName('NUMWAY').AsString:=Copy(SMess[i], 6, 2);
                qProgrRospusk.FieldByName('QUANCARS').AsInteger:=StrToInt(Copy(SMess[i], 11, 2));
                qProgrRospusk.FieldByName('CHECKROL').AsString:=Copy(SMess[i], 20, 1);
                qProgrRospusk.Post;
                flag_ins:=true;
                i:=i+1;
                end;
              qTrainPP.Edit;
              qTrainPP.FieldByName('DT_INPUTSL').AsString:=DateTimeToStr(Now);
              qTrainPP.FieldByName('PROGRROSP_YES').AsString:='1';
              if qTrainPPNumTrain.Value='' then
                qTrainPPNumTrain.Value:=NumTrain;
              qTrainPP.Post;
              frMyShowMess.lMess.Font.Color:=clBlue;
              frMyShowMess.lMess.Caption:='Отримано сортувальний лист!!!';
              frMyShowMess.Show;
              end
            End;
          CloseFile(F);
          if RospuskTrain<>0 then
            with qTrainPP do
            locate('TrainPP_Id', RospuskTrain, [loCaseInsensitive]);
        End;
      End;
      End;
  end;
end;

procedure TfrMain.tmSortListTimer(Sender: TObject);
Var F:TextFile;
  SMess:ARRAY[1..1000] of String;
  NumTrain, NumWay, tt:String;
  n_str,i, m:integer;
  flagOk:boolean;
begin
//   SetCurrentDir('d:\Arm\InData\');
  flbMessKSEOD.Directory:='d:\Arm\files32\locuseri\';
  flbMessKSEOD.Update;
  flbMessKSEOD.ItemIndex:=0;
  flagOk:=false;
 if (flbMessKSEOD.Items[0]<>'[..]') and (qProgrRospusk.State<>dsInsert) and (qProgrRospusk.State<>dsEdit) then
  begin
  AssignFile(F, flbMessKSEOD.Items[0]);
  Reset(F);
  i:=0;
  while not eof(F) do
```

```
  Begin
  i:=i+1;
  readln(F, SMess[i]);
  End;
 n_str:=i;
 if (Copy(SMess[1], 1, 5)='(:201') and (Copy(SMess[1],Pos('/',SMess[1])-1,1)='2') then
  begin
  with qTrainPP do
   if (locate('NumWay', IntToStr(StrToInt(Copy(SMess[1],Pos('/',SMess[1])+1,2)))+'IT', [loCaseInsensitive])) and
    (Copy(SMess[1],Pos('/',SMess[1])-1,1)='2') then
    Begin
    qTrainPP.Edit;
    qTrainPP.FieldByName('NumTrain').AsString:=Copy(SMess[1], 13, 4);
    qTrainPP.Post;
    End
  end
 else
  Begin
  for i:=1 to n_str-1 do
   Begin
   if (Copy(SMess[i], Length(SMess[i])-4, 5)='02/01') or
    (Copy(SMess[i], Length(SMess[i])-4, 5)='02/02') or
    (Copy(SMess[i], Length(SMess[i])-4, 5)='02/03') or
    (Copy(SMess[i], Length(SMess[i])-4, 5)='02/04') or
    (Copy(SMess[i], Length(SMess[i])-4, 5)='02/05') or
    (Copy(SMess[i], Length(SMess[i])-4, 5)='02/07') or
    (Copy(SMess[i], Length(SMess[i])-4, 5)='02/22') or
    (Copy(SMess[i], Length(SMess[i])-4, 5)='02/29') or
    (Copy(SMess[i], Length(SMess[i])-4, 5)='02/30') or
    (Copy(SMess[i], Length(SMess[i])-4, 5)='02/52') or
    (Copy(SMess[i], Length(SMess[i])-4, 5)='02/56') or
    (Copy(SMess[i], Length(SMess[i])-4, 5)='02/68') then
    flagOk:=true;
   End;
  if  (n_str>10) and flagOk then
   Begin
   i:=1;
//    while (Length(SMess[i])<>8) and (i<n_str) do
   while (Copy(SMess[i], 1, 3)<>' 01') and (i<n_str) do
    i:=i+1;
   NumTrain:=Copy(SMess[i-3], 1, 4);
   tt:=Copy(SMess[i-3], Length(SMess[i-3])-1, 2);
   NumWay:=IntToStr(StrToInt(Copy(SMess[i-3], Length(SMess[i-3])-1, 2)));
   with qTrainPP do
    if locate('NumWay', NumWay+'IT', [loCaseInsensitive]) then
     Begin
     if qTrainPPPROGRROSP_YES.Value='1' then
      begin
      qProgrRospusk.First;
      while not qProgrRospusk.eof do
       qProgrRospusk.delete;
      qProgrRospusk.Close;
      qProgrRospusk.Open;
```

```
            qSortList.first;
            while not qSortList.eof do
              qSortList.delete;
            qSortList.Close;
            qSortList.Open;
            locate('NumWay', NumWay+'П', [loCaseInsensitive]);
            end;
//          i:=i+1;
           while (i<n_str) and (SMess[i]<>'') and (Copy(SMess[i], 1, 1)=' ') do
            Begin
            qSortList.Insert;
            qSortList.FieldByName('TrainPP_Id').AsInteger:=qTrainPPTrainPP_Id.Value;
            qSortList.FieldByName('NUMOTCEPA').AsInteger:=StrToInt(Copy(SMess[i], 2, 2));
            qSortList.FieldByName('NUMWAY').AsString:=Copy(SMess[i], 6, 2);
            qSortList.FieldByName('QUANCARS').AsInteger:=StrToInt(Copy(SMess[i], 11, 2));
            qSortList.FieldByName('CHECKROL').AsString:=Copy(SMess[i], 20, 1);
            qSortList.FieldByName('WEIGHTOTCEPA').AsString:=Copy(SMess[i], 15, 4);
            qSortList.FieldByName('NUMFIRSTCAR').AsString:=Copy(SMess[i], 28, 8);   {!!!}
            qSortList.Post;
            flag_ins:=false;
            qProgrRospusk.Insert;
            qProgrRospusk.FieldByName('TrainPP_Id').AsInteger:=qTrainPPTrainPP_Id.Value;
            qProgrRospusk.FieldByName('NUMOTCEPA').AsInteger:=StrToInt(Copy(SMess[i], 2, 2));
            qProgrRospusk.FieldByName('NUMWAY').AsString:=Copy(SMess[i], 6, 2);
            qProgrRospusk.FieldByName('QUANCARS').AsInteger:=StrToInt(Copy(SMess[i], 11, 2));
            qProgrRospusk.FieldByName('CHECKROL').AsString:=Copy(SMess[i], 20, 1);
            qProgrRospusk.Post;
            flag_ins:=true;
            i:=i+1;
            end;
           qTrainPP.Edit;
           qTrainPP.FieldByName('DT_INPUTSL').AsString:=DateTimeToStr(Now);
           qTrainPP.FieldByName('PROGRROSP_YES').AsString:='1';
           if qTrainPPNumTrain.Value='' then
            qTrainPPNumTrain.Value:=NumTrain;
           qTrainPP.Post;
           frMyShowMess.lMess.Font.Color:=clBlue;
           frMyShowMess.lMess.Caption:='Получен сортировочный лист!!!';
           frMyShowMess.Show;
           end
        End;
      End;
    CloseFile(F);
    Erase(F);
    if RospuskTrain<>0 then
      with qTrainPP do
      locate('TrainPP_Id', RospuskTrain, [loCaseInsensitive]);
    End;
end;

procedure SendKontr(var StrSend:string);
begin
//  frMain.Timer1.Interval:= 200;
```

```
  frMain.Timer1.Enabled:= true;
  MesSend:= StrSend;
end;

procedure TfrMain.FormActivate(Sender: TObject);
begin
  qSortList.Close;
  qSortList.Open;
  qProgrRospusk.Close;
  qProgrRospusk.Open;
end;

procedure TfrMain.sbStartClick(Sender: TObject);
var StrSend:string;
    err:boolean;
    i, sum_code:integer;
begin
if MessageDlg('Завантажити програму розпуску в контролер?',
    mtConfirmation, [mbYes, mbNo], 0) = mrYes  then
Begin
if sbStart.Enabled=false then Exit;
err:=false;
qProgrRospusk.First;
While not qProgrRospusk.Eof do
  if not qProgrRospusk.eof then
    begin
    if (StrToInt(qProgrRospuskNUMWAY.Value)<3) or (StrToInt(qProgrRospuskNUMWAY.Value)>18) or
       (qProgrRospuskQuanCars.Value<1) then
      Begin
      err:=true;
      frMyShowMess.lMess.Caption:='Помилка даних у відчепі номер - '+IntToStr(qProgrRospuskNumOtcepa.Value);
      frMyShowMess.Show;
      end;
    StrSend:='';
    Case Length(IntToStr(qProgrRospuskNumOtcepa.Value)) of
         1: StrSend:=StrSend+' 00'+IntToStr(qProgrRospuskNumOtcepa.Value);
         2: StrSend:=StrSend+' 0'+IntToStr(qProgrRospuskNumOtcepa.Value);
         3: StrSend:=StrSend+' '+IntToStr(qProgrRospuskNumOtcepa.Value);
    end;
    if Length(qProgrRospuskNUMWAY.Value)=1 then
      StrSend:=StrSend+' 0'+qProgrRospuskNUMWAY.Value
    else
      StrSend:=StrSend+' '+qProgrRospuskNUMWAY.Value;
    if qProgrRospuskQuanCars.Value<20 then
      Begin
      if qProgrRospuskQuanCars.Value<10 then
        StrSend:=StrSend+' 0'+IntToStr(qProgrRospuskQuanCars.Value)+':)'
      else
        StrSend:=StrSend+' '+IntToStr(qProgrRospuskQuanCars.Value)+':)';
      end
     else
        StrSend:=StrSend+' '+'19'+':)';
    sum_code:=0;
```

```
    for i:=1 to Length(StrSend)-2 do
     if Copy(StrSend,i,1)<>' ' then
      sum_code:=sum_code+StrToInt(Copy(StrSend,i,1));
//    sum_code:=sum_code+1;
    Case Length(IntToStr(sum_code)) of
         1: StrSend:='(:01 000'+IntToStr(sum_code)+StrSend;
         2: StrSend:='(:01 00'+IntToStr(sum_code)+StrSend;
         3: StrSend:='(:01 0'+IntToStr(sum_code)+StrSend;
         4: StrSend:='(:01 '+IntToStr(sum_code)+StrSend;
     end;
    mProgrRospusk.Lines.Add(StrSend);
    StrSend:='';
    qProgrRospusk.Next;
    end;
    mProgrRospusk.Lines.Add('(:01 0000 000 00 00:)');
    if err then
     begin
     while mProgrRospusk.Lines.Count<>0 do
      mProgrRospusk.Lines.Delete(0);
     end
    else
     begin
     sbStart.Enabled:=false;
     sbContine.Enabled:=false;
//     gTrainPP.Enabled:=false;
     RospuskTrain:=qTrainPPTrainPP_Id.Value;
     RospuskTrainNum:=qTrainPPNumTrain.Value;
     PrRospuskBe:=true;
     shReady.Brush.Color:=clRed;
     lReady.Caption:='  Завантажено програму розпуску');
     end;
end;
end;

procedure TfrMain.gProgrRospuskKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
Var QuanCars, TekCode, NumOtc, NextCar:integer;
  CR:String;
begin
  QuanCars:=0;
  Case Key of
    VK_F3: begin
         TekCode:=qProgrRospuskProgrRospusk_Id.Value;
         if qProgrRospusk.FindNext then
           Begin
           QuanCars:=qProgrRospuskQuanCars.Value;
           qProgrRospusk.Delete;
           if TekCode<>qProgrRospuskProgrRospusk_Id.Value then
             qProgrRospusk.Prior;
           qProgrRospusk.Edit;
           qProgrRospuskQuanCars.Value:=qProgrRospuskQuanCars.Value+QuanCars;
           qProgrRospusk.Post;
           NumOtc:=qProgrRospuskNumOtcepa.Value+1;
```

```
          qProgrRospusk.Next;
          while not qProgrRospusk.eof do
           begin
           qProgrRospusk.Edit;
           qProgrRospuskNumOtcepa.Value:=NumOtc;
           qProgrRospusk.Post;
           NumOtc:=NumOtc+1;
           qProgrRospusk.Next;
           end;
           with qProgrRospusk do
            locate('ProgrRospusk_Id', TekCode, [loCaseInsensitive]);
          qProgrRospuskNumWay.FocusControl;
          qProgrRospusk.Edit;
          End
         else
          ShowMessage('Останній відчіп. Помилка об"єднання!!!');
         end;
   VK_F4: Begin
         frQuanCar.Show;
{        NumOtc:=qProgrRospuskNumOtcepa.Value;
         CR:=qProgrRospuskCheckRol.Value;
         qProgrRospusk.Last;
         NextCar:=qProgrRospuskNumOtcepa.Value+1;
         while qProgrRospuskNumOtcepa.Value>NumOtc do
          Begin
          qProgrRospusk.Edit;
          qProgrRospuskNumOtcepa.Value:=NextCar;
          qProgrRospusk.Post;
          NextCar:=NextCar-1;
          qProgrRospusk.Prior;
          End;
         flag_ins:=false;
         qProgrRospusk.Insert;
         qProgrRospusk.FieldByName('TrainPP_Id').AsInteger:=qTrainPPTrainPP_Id.Value;
         qProgrRospusk.FieldByName('NumOtcepa').AsInteger:=NumOtc+1;
         qProgrRospusk.FieldByName('NumWay').AsString:='00';
         qProgrRospusk.FieldByName('QuanCars').AsInteger:=0;
         qProgrRospusk.FieldByName('CheckRol').AsString:=CR;
         qProgrRospusk.Post;
         flag_ins:=true;
         with qProgrRospusk do
          Locate('NumOtcepa', NumOtc, [loCaseInsensitive]);
         OldQuanCars:=qProgrRospuskQuanCars.Value;
         qProgrRospuskQuanCars.FocusControl;
         Flag_Dobl:=true;
         qProgrRospusk.Edit;}
         End;
   13:   begin
         if qProgrRospusk.State=dsEdit then
          qProgrRospusk.Post;
         end;
  end;
end;
```

```
procedure TfrMain.SpeedButton6Click(Sender: TObject);
begin
  frAbout.Show;
end;

procedure TfrMain.sbF6Click(Sender: TObject);
begin
  ReportSortList.Preview;
end;

procedure TfrMain.tmContrReadyTimer(Sender: TObject);
begin
  tmContrReady.Enabled:=false;
  if ContrError then
    Begin
    frMyShowMess.lMess.Font.Color:=clRed;
    frMyShowMess.lMess.Caption:='Збій у контролері!!!';
    frMyShowMess.Show;
    shJobKontr.Brush.Color:=clRed;
    lContr.Caption:='Контролер несправний!';
    while mProgrRospusk.Lines.Count<>0 do
      mProgrRospusk.Lines.Delete(0);
    ContrError:=false;
    gTrainPP.Enabled:=true;
    PrRospuskBe:=false;
//    delDataProgrRosp;
    End;
end;

procedure TfrMain.tmWaitTimer(Sender: TObject);
begin
  tmWait.Enabled:=false;
  SendKontr(MesIn);
end;

procedure TfrMain.dsTrainPPDataChange(Sender: TObject; Field: TField);
begin
  if (qTrainPPPROGRROSP_YES.Value='1') and (shReady.Brush.Color<>clRed) and (shReady.Brush.Color<>clGreen)and
     (shJobKontr.Brush.Color<>clRed) and (SendPrRospusk=false) then
    begin
    sbStart.Enabled:=true;
    sbContine.Enabled:=true
    end
  else
    begin
    sbStart.Enabled:=false;
    sbContine.Enabled:=false;
    end;
end;

procedure TfrMain.sbF1Click(Sender: TObject);
begin
```

```
  frHelp.Show;
end;

procedure TfrMain.qProgrRospuskAfterEdit(DataSet: TDataSet);
begin
  CodeProgrRospusk:=qProgrRospuskProgrRospusk_Id.Value;
end;

procedure TfrMain.ServerSocket1ClientRead(Sender: TObject;
  Socket: TCustomWinSocket);
var CodeOper:integer;
  BlockInData:String;
begin
// SockContr:= Socket;
 TimerLive.Enabled:=false;
 TimerLive.Interval:=5000;
 TimerLive.Enabled:=true;
  BlockInData:=Socket.ReceiveText;
  if ((BlockInData[1]+BlockInData[2])<>'(:') or ((BlockInData[5]+BlockInData[6])<>':)') then
    Begin
    StoreData(BlockInData, 'Контролер', 'Помилка передачі!!!');
    MesIn:='(:09:)';
//              tmWait.Enabled:=true;
    SendKontr(MesIn);
    Abort;
    End;
    CodeOper:=StrToInt(BlockInData[3]+BlockInData[4]);   //Виділення коду повідомлення
    shJobKontr.Brush.Color:=clGreen;
    lContr.Caption:='Контролер справний';
Case CodeOper of
  13: Begin
      if errCShow then
        begin
        frMyShowMess.Close;
        errCShow:=false;
        End;
      End;
  2: Begin
           if frMyShowMess.lMess.Caption='Контролер не відповідає!!!' then
             frMyShowMess.Close;
           tmContrReady.Enabled:=false;
           ContrError:=false;
           tmContrReady.Interval:=2000;
           shJobKontr.Brush.Color:=clGreen;
           lContr.Caption:='Контролер справний';
           shReady.Brush.Color:=clYellow;
           lReady.Caption:='  Готовий прийняти програму розпуску');
           shMode.Brush.Color:=clGreen;
           lMode.Caption:='Автоматичний режим';
           if qTrainPPPROGRROSP_YES.Value='1' then
             begin
             sbStart.Enabled:=true;
             sbContine.Enabled:=true;
```

```
          end;
        if not PrRospuskBe then
          begin
          MesIn:='(:03:)';
//          tmWait.Enabled:=true;
          SendKontr(MesIn);
          end
        else
          Begin
          shReady.Brush.Color:=clGreen;
          lReady.Caption:='  Завантажено програму розпуску';
          NumOtc:=0;
          MesIn:=mProgrRospusk.Lines.Strings[NumOtc];
          StoreData(MesIn, 'АРМ', 'Програма розпуску');
//          tmWait.Enabled:=true;
          SendKontr(MesIn);
          NumOtc:=NumOtc+1;
          SendPrRospusk:=true;
          sbStart.Enabled:=false;
          sbContine.Enabled:=false;
          End;
        ContrError:=true;
        tmContrReady.Enabled:=true;
    End;
  3: Begin
        shReady.Brush.Color:=clRed;
        lReady.Caption:='  Не готовий прийняти програму розпуску';
        ContrError:=false;
        MesIn:='(:10:)';
//          tmWait.Enabled:=true;
        SendKontr(MesIn);
        End;
  11: Begin
        StoreData(BlockInData, 'Контролер', 'Перехід у ручний режим');
        shMode.Brush.Color:=clBlue;
        lMode.Caption:='Ручний режим';
        shReady.Brush.Color:=clRed;
        lReady.Caption:='  Не готовий прийняти програму розпуску';
        ContrError:=false;
        MesIn:='(:10:)';
        StoreData(MesIn, 'АРМ', 'Зрозумів');
//          tmWait.Enabled:=true;
        SendKontr(MesIn);
        gTrainPP.Enabled:=true;
        End;
  15: Begin
        StoreData(BlockInData, 'Контролер', 'Перехід в автоматичний режим');
        shMode.Brush.Color:=clGreen;
        ContrError:=false;
        lMode.Caption:='Автоматичний режим';
        MesIn:='(:10:)';
        StoreData(MesIn, 'АРМ', 'Зрозумів');
//          tmWait.Enabled:=true;
```

```
          SendKontr(MesIn);
          End;
  16: Begin
// Нажата кнопка "сброс"
          ContrError:=false;
          StoreData(BlockInData, 'Контролер', 'Натиснуто кнопку "Сброс"');
          MesIn:='(:10:)';
          StoreData(MesIn, 'АРМ', 'Зрозумів');
          SendKontr(MesIn);
          End;
  10: Begin
//Понял
          QuanOtv:=0;
          tmContrReady.Enabled:=false;
          ContrError:=false;
           if SendPrRospusk then
            Begin
            MesIn:=mProgrRospusk.Lines.Strings[NumOtc];
            StoreData(MesIn, 'АРМ', 'Програма розпуску');
//           tmWait.Enabled:=true;
            SendKontr(MesIn);
            NumOtc:=NumOtc+1;
            ContrError:=true;
            tmContrReady.Enabled:=true;
            End
          else
            MesIn:='';
          if MesIn='(:01 0000 000 00 00:)' then
            Begin
            while mProgrRospusk.Lines.Count<>0 do
              mProgrRospusk.Lines.Delete(0);
            ContrError:=true;
            PrRospuskBe:=false;
            SendPrRospusk:=false
            End;
      End;
  9: Begin
// Не понял
    gTrainPP.Enabled:=true;
          QuanOtv:=QuanOtv+1;
          if QuanOtv>5 then
            Begin
            frMyShowMess.lMess.Font.Color:=clRed;
            frMyShowMess.lMess.Caption:='Збій у контролері!!! Перхід у ручний режим';
            frMyShowMess.Show;
            MesIn:='(:01 0000 000 00 00:)';
            SendPrRospusk:=false;
            PrRospuskBe:=false;
            while mProgrRospusk.Lines.Count<>0 do
              mProgrRospusk.Lines.Delete(0);
//            tmWait.Enabled:=true;
            SendKontr(MesIn);
            End
```

```
          else
           begin
           tmContrReady.Enabled:=false;
           ContrError:=false;
           tmContrReady.Interval:=2000;
           StoreData(BlockInData, 'Контролер', 'Не зрозумів, повтори');
           MesIn:=mProgrRospusk.Lines.Strings[NumOtc-1];
           StoreData(MesIn, 'АРМ', 'Програма розпуску (повтор)');
//           tmWait.Enabled:=true;
           SendKontr(MesIn);
           ContrError:=true;
           tmContrReady.Enabled:=true;
           end;
          End;
  17: Begin
// Нажата кнопка "Продвижение"
          ContrError:=false;
          StoreData(BlockInData, 'Контролер', 'Натиснуто кнопку "Просування"');
          MesIn:='(:10:)';
          StoreData(MesIn, 'АРМ', 'Зрозумів');
          SendKontr(MesIn);
          End;
  18: Begin
//Нажата кнопка "задержка"
          ContrError:=false;
          StoreData(BlockInData, 'Контролер', 'Натиснуто кнопку "Затримка"');
          MesIn:='(:10:)';
          StoreData(MesIn, 'АРМ', 'Понял');
          SendKontr(MesIn);
          End;
  20: Begin
//Конец роспуска состава
          ContrError:=false;
          tmContrReady.Enabled:=false;
          StoreData(BlockInData, 'Контролер', 'Кінець розпуску составу');
          shMode.Brush.Color:=clBlue;
          lMode.Caption:='Ручний режим';
          MesIn:='(:10:)';
          StoreData(MesIn, 'АРМ', 'Зрозумів');
          SendKontr(MesIn);
          tmContrReady.Enabled:=true;
          gTrainPP.Enabled:=true;
          sbStart.Enabled:=true;
          sbContine.Enabled:=true;
          frMyShowMess.lMess.Font.Color:=clBlue;
          frMyShowMess.lMess.Caption:='Розпуск завершено!!!';
          frMyShowMess.Show;
          PrRospuskBe:=false;
          delDataProgrRosp;
          End;
  21: Begin
//Кінець розпуску составу в автоматичному режимі
          ContrError:=false;
```

```
          tmContrReady.Enabled:=false;
          StoreData(BlockInData, 'Контролер', 'Розпуск в автоматичному режимі завершено');
          MesIn:='(:10:)';
          StoreData(MesIn, 'АРМ', 'Зрозумів');
          SendKontr(MesIn);
          tmContrReady.Enabled:=true;
//          lReady.Caption:='  Скинуто програму розпуску');
          shReady.Brush.Color:=clRed;
          gTrainPP.Enabled:=true;
//          sbStart.Enabled:=false;
          frMyShowMess.lMess.Font.Color:=clBlue;
          frMyShowMess.lMess.Caption:='Розпуск в автоматичному режимі завершено!!!';
          frMyShowMess.Show;
          PrRospuskBe:=false;
//          delDataProgrRosp;
          End;
else
// Не понял, повтори
          Begin
          ContrError:=false;
          StoreData(BlockInData, 'Контролер', 'Код не розшифровано АРМом');
          MesIn:='(:09:)';
          SendKontr(MesIn);
          StoreData(MesIn, 'АРМ', 'Зрозумів');
          End;
end;
  BlockInData:='';
end;

procedure TfrMain.Timer1Timer(Sender: TObject);
var j, ConnNum: Integer;
begin
  Timer1.Enabled:= false;
{  if (frMain.ServerSocket1.Socket.Connected) and
     (frMain.ServerSocket1.Socket.RemoteAddress='192.168.1.106') then
    frMain.ServerSocket1.Socket.Connections[0].SendText(MesSend)
  else
    MessageDlg('Надіслати відповідь не вдалося', mtConfirmation, [mbOK, mbNo], 0);}
  with frMain.ServerSocket1.Socket do
  begin
    ConnNum:=-1;
    WaitConnFlag:= false;
    TimWaitConn.Enabled:= true; {6000мс = 6с}
    repeat
      Label1.Caption:=(inttostr(ActiveConnections));
    until Connected or WaitConnFlag; {з'єднався клієнт або спливло 6с}
    TimWaitConn.Enabled:= false;
    if (not Connected) then
    begin
      MessageDlg('Немає з"єднання', mtConfirmation, [mbOK], 0);
      exit;
    end;
    for j:=0 to (ActiveConnections-1) do
```

```
    if Connections[j].RemoteAddress='192.168.1.106' then ConnNum:=j;
   if ConnNum<>-1
    then Connections[ConnNum].SendText(MesSend)
    else MessageDlg('Отправить ответ не удалось'+inttostr(ActiveConnections), mtConfirmation, [mbOK], 0);
  end;
end;

{Поцедура закриває всі з'єднання з контролером
забезпечення можливості подальшого коректного підключення}
procedure ConnClose;
var j: integer;
begin
  with frMain.ServerSocket1.Socket do
  begin
   frMain.Label1.Caption:=(inttostr(ActiveConnections));
   if Connected then
   for j:=0 to (ActiveConnections-1) do
    if Connections[j].RemoteAddress='192.168.1.106' then Connections[j].Close;
   frMain.Label3.Caption:=(inttostr(ActiveConnections));
  end;
end;

procedure TfrMain.TimerLiveTimer(Sender: TObject);
begin
   ConnClose; //потрібно для зв'язку по TCP
   errCShow:=true;
   shJobKontr.Brush.Color:=clRed;
   lContr.Caption:='Контролер несправний';
   TimerLive.Enabled:=false;
   frMyShowMess.lMess.Font.Color:=clRed;
   frMyShowMess.lMess.Caption:='Контролер не відповідає!!!';
   frMyShowMess.Show;
end;

procedure TfrMain.TimWaitConnTimer(Sender: TObject);
begin
  TimWaitConn.Enabled:= false;
  WaitConnFlag:= true;
end;

procedure TfrMain.sbClearClick(Sender: TObject);
begin
if MessageDlg('Очистити колію?',
   mtConfirmation, [mbYes, mbNo], 0) = mrYes  then
  begin
  RospuskTrain:=qTrainPPTrainPP_Id.Value;
  delDataProgrRosp;
  end;
end;

procedure TfrMain.sbContineClick(Sender: TObject);
var i, sum_code, NumOtcCon:integer;
  err:boolean;
```

```
  StrSend:string;
begin
if MessageDlg('Завантажити програму розпуску у контролер? Відчіп - '+
            IntToStr(qProgrRospuskNumOtcepa.Value),
   mtConfirmation, [mbYes, mbNo], 0) = mrYes  then
Begin
if sbContine.Enabled=false then Exit;
err:=false;
NumOtcCon:=1;
While not qProgrRospusk.Eof do
  if not qProgrRospusk.eof then
    begin
    if (StrToInt(qProgrRospuskNUMWAY.Value)<3) or (StrToInt(qProgrRospuskNUMWAY.Value)>18) or
      (qProgrRospuskQuanCars.Value<1) then
     Begin
     err:=true;
     frMyShowMess.lMess.Caption:='Помилка даних відчіп номер - '+IntToStr(qProgrRospuskNumOtcepa.Value);
     frMyShowMess.Show;
     end;
    StrSend:='';
    Case Length(IntToStr(NumOtcCon)) of
         1: StrSend:=StrSend+' 00'+IntToStr(NumOtcCon);
         2: StrSend:=StrSend+' 0'+IntToStr(NumOtcCon);
         3: StrSend:=StrSend+' '+IntToStr(NumOtcCon);
    end;
    if Length(qProgrRospuskNUMWAY.Value)=1 then
     StrSend:=StrSend+' 0'+qProgrRospuskNUMWAY.Value
    else
     StrSend:=StrSend+' '+qProgrRospuskNUMWAY.Value;
    if qProgrRospuskQuanCars.Value<20 then
     Begin
     if qProgrRospuskQuanCars.Value<10 then
       StrSend:=StrSend+' 0'+IntToStr(qProgrRospuskQuanCars.Value)+':)'
     else
       StrSend:=StrSend+' '+IntToStr(qProgrRospuskQuanCars.Value)+':)';
     end
    else
       StrSend:=StrSend+' '+'19'+':)';
    sum_code:=0;
    for i:=1 to Length(StrSend)-2 do
     if Copy(StrSend,i,1)<>' ' then
       sum_code:=sum_code+StrToInt(Copy(StrSend,i,1));
//    sum_code:=sum_code+1;
    Case Length(IntToStr(sum_code)) of
         1: StrSend:='(:01 000'+IntToStr(sum_code)+StrSend;
         2: StrSend:='(:01 00'+IntToStr(sum_code)+StrSend;
         3: StrSend:='(:01 0'+IntToStr(sum_code)+StrSend;
         4: StrSend:='(:01 '+IntToStr(sum_code)+StrSend;
    end;
    mProgrRospusk.Lines.Add(StrSend);
    StrSend:='';
    NumOtcCon:=NumOtcCon+1;
    qProgrRospusk.Next;
```

```
    end;
    mProgrRospusk.Lines.Add('(:01 0000 000 00 00:)');
    if err then
     begin
     while mProgrRospusk.Lines.Count<>0 do
       mProgrRospusk.Lines.Delete(0);
     end
    else
     begin
     sbStart.Enabled:=false;
     sbContine.Enabled:=false;
//     gTrainPP.Enabled:=false;
     RospuskTrain:=qTrainPPTrainPP_Id.Value;
     RospuskTrainNum:=qTrainPPNumTrain.Value;
     PrRospuskBe:=true;
     shReady.Brush.Color:=clRed;
     lReady.Caption:='  Завантажено програму розпуску');
     end;
end;
end;

procedure TfrMain.dsProgrRospuskUpdateData(Sender: TObject);
begin
  if (qProgrRospusk.State=dsInsert) and (flag_ins) then
    qProgrRospusk.Cancel;
end;

end.
```

A.2 Лістінг About.pas

```
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, Buttons, ExtCtrls;

type
  TfrAbout = class(TForm)
    GroupBox4: TGroupBox;
    Label11: TLabel;
    BitBtn1: TBitBtn;
    Label1: TLabel;
    procedure BitBtn1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frAbout: TfrAbout;
```

```
implementation

{$R *.DFM}

procedure TfrAbout.BitBtn1Click(Sender: TObject);
begin
  Close;
end;

end.
```

А.3 Лістінг Help.pas

```
unit Help;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls;

type
  TfrHelp = class(TForm)
    mmHelp: TMemo;
    procedure FormShow(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frHelp: TfrHelp;

implementation

{$R *.DFM}

procedure TfrHelp.FormShow(Sender: TObject);
begin
  mmHelp.Lines.LoadFromFile('d:\Arm\FHelp.doc');
end;

end.
```

А.4 Лістінг MyShowMess.pas

```
unit MyShowMess;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, Buttons, ExtCtrls;

type
  TfrMyShowMess = class(TForm)
    BitBtn1: TBitBtn;
    lMess: TLabel;
    TimerMess: TTimer;
    procedure BitBtn1Click(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure TimerMessTimer(Sender: TObject);
    procedure FormShow(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frMyShowMess: TfrMyShowMess;

implementation

uses Main;

{$R *.DFM}

procedure TfrMyShowMess.BitBtn1Click(Sender: TObject);
begin
 Close;
end;

procedure TfrMyShowMess.FormClose(Sender: TObject;
  var Action: TCloseAction);
begin
  TimerMess.Enabled:=false;
  lMess.Caption:='';
  frMain.FocusControl( frMain.Panel1);
end;

procedure TfrMyShowMess.TimerMessTimer(Sender: TObject);
begin
  lMess.Visible:=not lMess.Visible;
  Beep;
```

```
end;

procedure TfrMyShowMess.FormShow(Sender: TObject);
begin
  TimerMess.Enabled:=true;
end;

end.
```

A.5 Лістинг Protocol.pas

```
unit Protocol;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, Buttons, DBCtrls, ExtCtrls, Db, DBTables, Grids, DBGrids,
  ComCtrls, Mask;

type
  TfrProtocol = class(TForm)
    Panel1: TPanel;
    DBNavigator1: TDBNavigator;
    BitBtn1: TBitBtn;
    DBGrid1: TDBGrid;
    qStoreData: TQuery;
    dsStoreData: TDataSource;
    SpeedButton1: TSpeedButton;
    qStoreDataSTOREDATA_ID: TIntegerField;
    qStoreDataDATEOPER: TDateTimeField;
    qStoreDataTIMEOPER: TDateTimeField;
    qStoreDataNUMTRAIN: TStringField;
    qStoreDataCODEOPER: TStringField;
    qStoreDataWHOSEND: TStringField;
    qStoreDataREMAR: TStringField;
    cbDate: TCheckBox;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    SelDate: TDateTimePicker;
    Shape1: TShape;
    Label4: TLabel;
    cbTrain: TCheckBox;
    SelTrain: TEdit;
    Shape2: TShape;
    Label5: TLabel;
    cbAll: TCheckBox;
    Shape3: TShape;
    SpeedButton2: TSpeedButton;
```

```
    SelTimeB: TDateTimePicker;
    SelTimeE: TDateTimePicker;
    procedure BitBtn1Click(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frProtocol: TfrProtocol;

implementation

uses RepProtocol, repProtocol1;

{$R *.DFM}

procedure TfrProtocol.BitBtn1Click(Sender: TObject);
begin
  Close;
end;

procedure TfrProtocol.SpeedButton1Click(Sender: TObject);
begin
  ReportProtocol.Preview;
end;

procedure TfrProtocol.FormCreate(Sender: TObject);
begin
  cbAll.Checked:=true;
  cbDate.Checked:=false;
  cbTrain.Checked:=false;
  SelDate.Date:=Now;
  SelTimeB.Time:=StrToTime('00:00:01');
  SelTimeE.Time:=StrToTime('23:59:59');
end;

procedure TfrProtocol.SpeedButton2Click(Sender: TObject);
begin
  qStoreData.Close;
  qStoreData.SQL.Clear;
  qStoreData.SQL.Add('SELECT * FROM StoreData');
  if cbDate.Checked=true then
    begin
    qStoreData.SQL.Add('WHERE (TimeOper>=:TB)');
    qStoreData.SQL.Add('and (TimeOper<=:TE)');
    if cbTrain.Checked=true then
      qStoreData.SQL.Add('and (NumTrain='+Char($27)+SelTrain.Text+Char($27)+')');
    qStoreData.Params[0].AsString:=DateToStr(SelDate.Date)+' '+TimeToStr(SelTimeB.Time);
```

```
    qStoreData.Params[1].AsString:=DateToStr(SelDate.Date)+' '+TimeToStr(SelTimeE.Time);
    qStoreData.Open;
    End
  else
    if cbTrain.Checked=true then
      Begin
      qStoreData.SQL.Add('WHERE (NumTrain='+Char($27)+SelTrain.Text+Char($27)+')');
      qStoreData.Open;
      End;
  if cbAll.Checked=true then
    Begin
    qStoreData.Close;
    qStoreData.SQL.Clear;
    qStoreData.SQL.Add('SELECT * FROM StoreData');
    qStoreData.Open;
    End;
end;

end.
```

А.6 Лістінг CuanCar.pas

```
unit QuanCar;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, db;

type
  TfrQuanCar = class(TForm)
    Label1: TLabel;
    edQuanCar: TEdit;
    Label2: TLabel;
    procedure edQuanCarKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure FormShow(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frQuanCar: TfrQuanCar;

implementation

uses Main;
```

```
{$R *.DFM}

procedure TfrQuanCar.edQuanCarKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
var
    QuanCars, NumOtc, NextCar:integer;
    CR:String;
a:integer;
begin
 case Key of
 13:
  try
    a:=StrToInt(edQuanCar.Text);
    with frMain do
     Begin
         OldQuanCars:=qProgrRospuskQuanCars.Value;
         NumOtc:=qProgrRospusk.FieldByName('NumOtcepa').AsInteger;
         CR:=qProgrRospuskCheckRol.Value;
         qProgrRospusk.Last;
         NextCar:=qProgrRospuskNumOtcepa.Value+1;
         while qProgrRospuskNumOtcepa.Value>NumOtc do
          Begin
          qProgrRospusk.Edit;
          qProgrRospuskNumOtcepa.Value:=NextCar;
          qProgrRospusk.Post;
          NextCar:=NextCar-1;
          qProgrRospusk.Prior;
          End;
         flag_ins:=false;
         qProgrRospusk.Insert;
         qProgrRospusk.FieldByName('TrainPP_Id').AsInteger:=qTrainPPTrainPP_Id.Value;
         qProgrRospusk.FieldByName('NumOtcepa').AsInteger:=NumOtc+1;
         qProgrRospusk.FieldByName('NumWay').AsString:='00';
         if frQuanCar.edQuanCar.Text='' then
          qProgrRospusk.FieldByName('QuanCars').AsInteger:=OldQuanCars
         else
          qProgrRospusk.FieldByName('QuanCars').AsInteger:=OldQuanCars-StrToInt(frQuanCar.edQuanCar.Text);
         qProgrRospusk.FieldByName('CheckRol').AsString:=CR;
         qProgrRospusk.Post;
         flag_ins:=true;
         with qProgrRospusk do
          Locate('NumOtcepa', NumOtc, [loCaseInsensitive]);
         qProgrRospusk.Edit;
         qProgrRospuskQuanCars.Value:=StrToInt(frQuanCar.edQuanCar.Text);
         qProgrRospusk.Post;
         qProgrRospusk.Close;
         qProgrRospusk.Open;
     End;
    Close;
    with frMain.qProgrRospusk do
          if Locate('NumOtcepa', NumOtc, [loCaseInsensitive]) then
         frMain.qProgrRospuskNumWay.FocusControl;
```

```
  except
    ShowMessage('Невірне введення даних');
  end;
 27: Begin
     edQuanCar.Text:='';
     Close;
     End;
 end;
end;

procedure TfrQuanCar.FormShow(Sender: TObject);
begin
edQuanCar.SetFocus;
edQuanCar.Text:='';
end;

end.
```

А.7 ЛістінгRepProtocol.pas

```
unit RepProtocol;

interface

uses Windows, SysUtils, Messages, Classes, Graphics, Controls,
  StdCtrls, ExtCtrls, Forms, Quickrpt, QRCtrls;

type
  TReportProtocol = class(TQuickRep)
    QRBand1: TQRBand;
    QRBand2: TQRBand;
    QRLabel7: TQRLabel;
    QRSysData2: TQRSysData;
    QRLabel2: TQRLabel;
    QRShape1: TQRShape;
    QRLabel3: TQRLabel;
    QRShape2: TQRShape;
    QRLabel4: TQRLabel;
    QRShape3: TQRShape;
    QRLabel5: TQRLabel;
    QRLabel6: TQRLabel;
    QRShape4: TQRShape;
    QRShape5: TQRShape;
    QRLabel1: TQRLabel;
  private

  public

  end;

var
  ReportProtocol: TReportProtocol;
```

```
implementation

{$R *.DFM}

end.
```

А.8 ЛістінгRepSortList.pas

```
unit RepSortList;

interface

uses Windows, SysUtils, Messages, Classes, Graphics, Controls,
  StdCtrls, ExtCtrls, Forms, Quickrpt, QRCtrls;

type
  TReportSortList = class(TQuickRep)
    QRBand1: TQRBand;
    QRBand2: TQRBand;
    QRBand3: TQRBand;
    QRDBText1: TQRDBText;
    QRLabel1: TQRLabel;
    QRLabel2: TQRLabel;
    QRLabel3: TQRLabel;
    QRLabel4: TQRLabel;
    QRLabel5: TQRLabel;
    QRLabel6: TQRLabel;
    QRShape1: TQRShape;
    QRShape2: TQRShape;
    QRShape3: TQRShape;
    QRShape4: TQRShape;
    QRShape5: TQRShape;
    QRDBText2: TQRDBText;
    QRDBText3: TQRDBText;
    QRDBText4: TQRDBText;
    QRDBText5: TQRDBText;
    QRDBText6: TQRDBText;
    QRLabel7: TQRLabel;
    QRLabel8: TQRLabel;
    QRLabel9: TQRLabel;
    QRLabel10: TQRLabel;
    QRSysData1: TQRSysData;
    QRDBText7: TQRDBText;
    QRDBText8: TQRDBText;
    QRDBText9: TQRDBText;
    QRDBText10: TQRDBText;
  private

  public

  end;
```

```
var
  ReportSortList: TReportSortList;

implementation

{$R *.DFM}

end.
```

А.9 Лістінг SQL - файл бази даних АРМу

```
DEFAULT CHARACTER SET WIN1251

/* Таблица поездов в парке прибытия */
CREATE TABLE TrainPP
(TrainPP_Id INTEGER NOT NULL PRIMARY KEY,
 DT_InputSL DATE,
 NumWay CHAR(2) NOT NULL,
 NumTrain CHAR(4),
 InfTrain CHAR(13),
 WeightTrain CHAR(5),
 LehgthTrain CHAR(4),
 DT_EndRospusk DATE
);

/* Сортировочный лист поезда*/
CREATE TABLE SortList
(SortList_Id INTEGER NOT NULL PRIMARY KEY,
 TrainPP_Id INTEGER NOT NULL REFERENCES TrainPP,
 NumOtcepa INTEGER NOT NULL,
 NumWay CHAR(2) NOT NULL,
 QuanCars INTEGER NOT NULL,
 CheckRol Char(1),
 WeightOtcepa CHAR(4),
 NumFirstCar CHAR(8) NOT NULL
);

CREATE INDEX SortList_In ON SortList (TrainPP_Id);

/* Программа роспуска поезда*/
CREATE TABLE ProgrRospusk
(ProgrRospusk_Id INTEGER NOT NULL PRIMARY KEY,
 TrainPP_Id INTEGER NOT NULL REFERENCES TrainPP,
 NumOtcepa INTEGER NOT NULL,
 NumWay CHAR(2) NOT NULL,
 QuanCars INTEGER NOT NULL,
 CheckRol Char(1)
);

CREATE INDEX ProgrRospusk_In ON ProgrRospusk (TrainPP_Id);
```

```
/* Сообщения обмена данными с контроллером и КС ЭОД*/
CREATE TABLE ExchangeData
(ExchangeData_Id INTEGER NOT NULL PRIMARY KEY,
 DT_Exchange DATE NOT NULL,
 In_Data CHAR(1),
 Out_Data CHAR(1),
 Data_Exchange BLOB NOT NULL
);

CREATE GENERATOR TrainPP_Gen;
CREATE GENERATOR SortList_Gen;
CREATE GENERATOR ProgrRospusk_Gen;
CREATE GENERATOR ExchangeData_Gen;

CREATE EXCEPTION NOT_VALID "NOT EDIT DATE";

SET TERM ###;

CREATE TRIGGER TrainPP_TR FOR TrainPP
ACTIVE BEFORE INSERT
AS BEGIN
  New.TrainPP_Id=GEN_ID(TrainPP_Gen,1);
END ###

CREATE TRIGGER SortList_TR FOR SortList
ACTIVE BEFORE INSERT
AS BEGIN
  New.SortList_Id=GEN_ID(SortList_Gen,1);
END ###

CREATE TRIGGER ProgrRospusk_TR FOR ProgrRospusk
ACTIVE BEFORE INSERT
AS BEGIN
  New.ProgrRospusk_Id=GEN_ID(ProgrRospusk_Gen,1);
END ###

CREATE TRIGGER ExchangeData_TR FOR ExchangeData
ACTIVE BEFORE INSERT
AS BEGIN
  New.ExchangeData_Id=GEN_ID(ExchangeData_Gen,1);
END ###

SET TERM ;###

COMMIT;

              EXIT;
```