РЕФЕРАТ

Пояснювальна записка: ___ с., ___ рис., ___ табл., __ дод., ___ джерел.

Об'єкт розробки: інформаційна система обробки та обліку повідомлень комерційного банку.

Мета кваліфікаційної роботи: автоматизація процесу збору статистичної інформації про роботу банку і на підставі цих даних створення повідомлень про сплату комісій третіх банків MT191, а також автоматизація процесу формування звітів та надання необхідних документів для проведення розслідувань та інших операцій з цінними паперами.

У вступі розглядається аналіз та сучасний стан проблеми, конкретизується мета кваліфікаційної роботи та галузь її застосування, наведено обґрунтування актуальності теми та уточнюється постановка завдання.

У першому розділі проведено аналіз предметної галузі, визначено актуальність завдання та призначення розробки, розроблена постановка завдання, задані вимоги до програмної реалізації, технологій та програмних засобів.

У другому розділі виконано аналіз існуючих рішень, обрано вибір платформи для розробки, виконано проектування і розробка програми, наведено опис алгоритму і структури функціонування програми, визначені вхідні і вихідні дані, наведені характеристики складу параметрів технічних засобів, описаний виклик та завантаження програми, описана робота програми.

В економічному розділі визначено трудомісткість розробленої інформаційної системи, проведений підрахунок вартості роботи по створенню програми та розраховано час на його створення.

Практичне значення роботи полягає в забезпеченні автоматизації процесу аналізу отриманих банком повідомлень, формування статистичних даних і на підставі цих даних створення повідомлення про сплату комісій третіх банків MT191. Система відправляє сформовані повідомлення банкам - відправникам MT103, з яких запитується комісія. Крім цього, на підставі отриманих статистичних даних, система визначає банки-неплатники з метою формування і відправки повторних запитів про необхідність сплати комісій і забезпечує формування щомісячних і щоквартальних звітів, які відображають отриманий прибуток відділу.

Актуальність роботи визначається великим попитом на подібні розробки, через те, що розроблене програмне забезпечення підвищує надійність і швидкість обробки фінансових повідомлень, а також знижує навантаження на персонал відділу шляхом автоматизації операторської роботи.

Список ключових слів: ІНФОРМАЦІЙНА СИСТЕМА, БАЗА ДАНИХ, КОМЕРЦІЙНИЙ БАНК, ПОВІДОМЛЕННЯ, РОЗРАХУНКИ.

ABSTRACT

Explanatory note: ___ pp., ___ fig., ___ table, __ appendix, ___ sources.

Object of development: information system for processing and accounting of commercial bank messages.

The purpose of the qualification work: automation of the process of collecting statistical information on the bank and on the basis of these data creating notifications of payment of commissions of third banks MT191, as well as automation of reporting and providing the necessary documents for investigations and other securities transactions.

The introduction considers the analysis and the current state of the problem, specifies the purpose of the qualification work and the scope of its application, provides a justification for the relevance of the topic and clarifies the problem.

In the first section the analysis of the subject area is carried out, the urgency of the task and purpose of development is defined, the statement of the task is developed, requirements to software realization, technologies and software are set.

The second section analyzes the existing solutions, selects the platform for development, performs design and development of the program, describes the algorithm and structure of the program, determines the input and output data, provides characteristics of the parameters of hardware, describes the call and download program, describes the program .

The economic section determines the complexity of the developed information system, calculates the cost of work to create a program and calculates the time for its creation.

The practical significance of the work is to ensure the automation of the process of analysis of messages received by the bank, the formation of statistical data and on the basis of these data to create a message about the payment of commissions of third banks MT191. The system sends the generated messages to the sending banks MT103, from which the commission is requested. In addition, based on the obtained statistics, the system identifies non-paying banks in order to form and send repeated requests for the need to pay commissions and ensures the formation of monthly and quarterly reports that reflect the profits of the department.

The urgency of the work is determined by the high demand for such developments, due to the fact that the developed software increases the reliability and speed of processing financial messages, as well as reduces the burden on the staff of the department by automating operator work.

List of keywords: INFORMATION SYSTEM, DATABASE, COMMERCIAL BANK, MESSAGES, CALCULATIONS.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД – база даних;

ІС – інформаційна система;

ООП – об’єктно-орієнтоване програмування;

ПЗ – програмне забезпечення;

СУБД – система управління базою даних;

BIC – Bank Identifier Codes, код ідентифікатора банку;

MT – Message Transaction, транзакція повідомлення;

SWIFT – Sosiety for Worldwide Interbank Financial Telecommunication, товариство всесвітніх банківських комунікацій.

ЗМІСТ

ВСТУП

Завдання автоматизації будь-якого процесу вимагає від фахівця чималих зусиль і кваліфікації. Це пояснюється тим, що для вирішення такого класу задач, необхідно досконально розуміти весь процес, який підлягає автоматизації. В даному випадку, від фахівця потрібні знання технологій баз даних, мережевих технологій, технологію обробки фінансових повідомлень і стандарти їх оформлення.

Формати стандартизованих машинопечатних повідомлень розроблені таким чином, щоб зробити їх найбільш незалежними від національних особливостей банківської сфери в кожній конкретній країні. У той же час уніфіковані формати повідомлень, що використовуються для передачі інформації в мережі SWIFT, поряд з присвоюваними суспільством банківськими ідентифікаційними кодами (восьмизначний код, який є унікальною адресою банківських та інших фінансових інститутів), рекомендовані Міжнародною організацією зі стандартизації (ISO) в якості міжнародних стандартів. Стандарти SWIFT стали стандартами «де факто» для фінансових повідомлень транзакцій, роблячи усе більший вплив на банківську справу різних країн.

Уніфікація машиночитаних форматів значно полегшує контроль коректності відправлених повідомлень, що, з одного боку забезпечує захист від випадкових помилок, і, з іншого - підвищує пропускну здатність системи для правильно сформульованих повідомлень. Процеси підготовки і обробки повідомлень повністю піддаються автоматизації, що значно підвищує ефективність і рентабельність банківської діяльності.

Система автоматичної обробки комісій третіх банків розроблена з метою підвищення продуктивності праці персоналу відділу шляхом автоматизації процесу обробки отриманої банком інформації, використовуючи сучасні програмно-технічні засоби. Розроблена система повинна забезпечити автоматизацію процесу аналізу отриманих банком повідомлень, формування

статистичних даних і на підставі цих даних створювати повідомлення про сплату комісій третіх банків MT191. Система повинна відправляти сформовані повідомлення банкам - відправникам MT103, з яких запитується комісія. Крім цього, на підставі отриманих статистичних даних, система повинна визначати банки-неплатники з метою формування і відправки повторних запитів про необхідність сплати комісій і забезпечити формування щомісячних і щоквартальних звітів, які відображають отриманий прибуток відділу.

Дана робота підвищує надійність і швидкість обробки фінансових повідомлень, а також знижує навантаження на персонал відділу шляхом автоматизації операторської роботи.

Розробка даної системи необхідна для швидкої і ефективної обробки повідомлень про кредитування MT103, формування повідомлень про сплату комісій третіх банків MT191, а також проводити пошук банків, які відмовляються сплачувати комісії, з метою розслідування і усунення причин таких відмов з повторним повідомленням банків про необхідність оплатити комісії.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1. Загальні відомості з предметної галузі

1.1.1. Міжнародна система S.W.I.F.T.

S.W.I.F.T. (Society for Worldwide Interbank Financial Telecommunications) - міжнародна міжбанківська система передачі інформації та здійснення платежів.

S.W.I.F.T. - кооперативне товариство, яке створене за бельгійським законодавством і належить його членам - більше 10 тисяч учасників з 210 країн (на жовтень 2019 року). Головний офіс знаходиться у Брюсселі.

Система переказів S.W.I.F.T. дозволяє здійснювати перекази в іноземній валюті у будь-яку країну світу на користь фізичних та юридичних осіб із використанням чи без використання рахунку. Перекази за системою S.W.I.F.T. ідеально підходять для оплати за покупки, які були здійснені за кордоном, бронювання готельних номерів, оплати навчання, відпочинку чи лікування, а також для грошових переказів родичам та друзям.

Існують два типи повідомлень:

– фінансові (безпосередньо між користувачами системи);
– системні (між користувачами та системою)..

Структурно всі повідомлення SWIFT складаються з:

1. Заголовку.
2. Тексту повідомлення.
3. Трейлеру.

Через комп'ютерний термінал (СВТ) здійснюється зв'язок з універсальним комп'ютером, передача і отримання повідомлень, а також управління прикладними завданнями. Повідомлення концентруються у регіональному процесорі(RGP), а потім перенаправляються на обробку до відповідного операційного центру. Там SWIFT обробляє повідомлення за таким алгоритмом:

– перевірка синтаксису;

– створення нових заголовків для перетворення повідомлень в початкову форму;

– додавання трейлерів;

– копіювання та шифрування повідомлень для зберігання.

В результаті перевірки, користувач отримує сповіщення: ACK - позитивний результат, NAK - негативний. Кожне повідомлення автоматично отримує вхідний номер.

Відправлення переказів за кордон здійснюється в євро, доларах США, російських рублях, англійських фунтах стерлінгів та швейцарських франках.

Виплата переказів із-за кордону здійснюється в євро, доларах США, російських рублях, англійських фунтах стерлінгів та швейцарських франках (також можлива конвертація в момент одержання готівки з рахунку).

Кошти доступні до виплати протягом 5 робочих днів після відправлення переказу.

Відправити гроші можна без відкриття рахунку (в сумі, що не перевищує в еквіваленті 400 000,00 грн/день), а з відкриттям рахунку - без обмежень щодо сум (із урахуванням вимог НБУ).

Для переказу коштів потрібен паспорт або документ, що його замінює а також реквізити одержувача (SWIFT-код банку отримувача, найменування банку отримувача, прізвище та ім'я або назву отримувача та його адресу, призначення платежу).

Наступні документи підтверджують підстави для відправлення SWIFT-переказу за межі України:

– договори (контракти);

– рахунки-фактури;

– позовні заяви, запрошення;

– листи-розрахунки або листи-повідомлення юридичних осіб-нерезидентів;

– листи адвокатів або нотаріусів іноземних держав.

Одержувачу переказу необхідно:

1. Звернутися у будь-яке відділення банку.
2. Пред’явити свій паспорт.
3. Після перевірки зазначених даних менеджером одержувачу буде виплачений переказ.

1.1.2. Опис стандартів SWIFT - повідомлень, оброблюваних системою

Категорії фінансових повідомлень:

1. Платежі та чеки нефінансових інститутів.
2. Трансфери фінансових інститутів.
3. Казначейські операції, форексні ринки, фінансові ринки.
4. Інкасо.
5. Ринки цінних паперів.
6. Ринки коштовностей, цінні метали.
7. Документарні акредитиви і гарантії.
8. Дорожні чеки.
9. Управління готівкою та статус учасника транзакції.
10. Повідомлення у вільному форматі.

У типі повідомлення категорію позначає перша цифра. Повідомлення всіх категорій, за винятком 3 і 9, проходять автоматичну аутентифікацію на справжність, тобто є "ключовими". Список банків, з якими ПриватБанк має такі ключі, регулярно пересилається відділом SWIFT зацікавленим підрозділам.

Поле повідомлення ідентифікується по своєму імені. Ім'я поля складається або з двох цифр, або з двох цифр й латинської літери верхнього регістру. Ім'я поля записується в новому рядку з першої позиції. Безпосередньо за ім'ям поля, йде роздільник ":", після якого записується зміст поля за стандартними форматами (наприклад,: 20 :,: 57A :). Будь-яке поле складається з обов'язкових підполів (частин) і необов'язкових (можливих). З тим, щоб відрізняти ступінь необхідності тієї чи іншої інформації, при визначенні

формату представлення даних, необов'язкова частина полягає в квадратні дужки [...]. Наприклад, якщо формат поля визначено як '16x [/ 4x]', то в цьому полі повинно бути присутнім до 16 символів. Наступні 4 символи, яким передує символ '/', необов'язкові, і тому вони можуть не бути присутніми в полі.

Кількість символів в рядку і кількість рядків в повідомленні визначається за допомогою наступного набору символів:

– nn - максимальна довжина;

– nn-nn - мінімальна і максимальна довжина;

– nn! - фіксована довжина;

– nn * nn - максимальна кількість рядків і максимальна довжина рядка (без урахування ідентифікатора поля).

Першим символом змісту будь-якого поля не можуть бути символи ':' і '-'. Дата записується у форматі YYMMDD, де YY - останні дві цифри року, MM - номер місяця (завжди дві цифри), DD - число місяця.

Правила запису чисел наступні:

– ціла частина повинна містити хоча б одну цифру;

– десяткова частина відділяється від цілої комою;

– максимальна довжина включає розділову кому;

– десяткову частину можна не писати, але розділова кома завжди має бути присутня (0,);

– в цілій частині числа дозволяються попередні нулі (000,00).

Дозволені типи символів:

– n - тільки цифри;

– a - літери алфавіту верхнього регістру;

– c - тільки цифри і букви алфавіту верхнього регістру;

– h-тільки шіснадцяткові букви від A до F (верхній регістр) і цифри;

– x - будь-який символ з дозволеного набору символів;

– e - пробіл;

– d - число.

Можливі способи ідентифікування (вказівки) учасників транзакції в

повідомленнях:

1. Міжнародний код банку (BIC-код).

Формат:

[/ 1a] [/ 34x] (ідентифікатор учасника)

4! A2! A2! C [3! C] (BIC - код)

Позначається опцією A відповідного поля (: 56A :,: 57A: ...).

2. Код відділення банку-відправника або банку-одержувача повідомлення.

Формат:

[/ 1a] [/ 34x] (ідентифікатор учасника)

[35x] (місцезнаходження)

Позначається опцією B відповідного поля (: 55B :,: 57B: ...).

3. Номер рахунку.

Формат:

/ 34x (рахунок)

Позначається опцією C відповідного поля (: 55C :,: 57C: ...).

4. Повне ім'я банку і адреса.

Формат:

[/ 1a] [/ 34x] (ідентифікатор учасника)

4 * 35x (назва та адреса)

Позначається опцією D відповідного поля (: 55D :,: 57D: ...).

До формування SWIFT повідомлень без помилок, слід ставитися дуже уважно, оскільки в мережі SWIFT, за кожне повідомлення з помилками формату стягує з банку-відправника вартість, що перевищує вартість звичайного повідомлення в п'ять разів.

1.2. Призначення розробки та галузь застосування

Призначенням даної інформаційної системи є формування SWIFT-повідомлень MT191 формату, ведення бази даних про кількість отриманих MT103 повідомлень і відправлених MT191 повідомлень, а також зберігання в допоміжній базі інформації про повідомлення наступних форматів:

– повідомлення MT910 (підтвердження кредитування);
– повідомлення MT191 (комісії банків);
– повідомлення MT202 (простий переклад фінансової організації);
– повідомлення MT950 (кредитне авізо);
– повідомлення MTn9n (вільний формат).

Інформаційна система chrgas призначена для формування повідомлень про сплату комісій третіх банків по кредитовим повідомленнями (повідомлення типу MT103), автоматизації процесу розслідувань по неоплачених комісіям з можливістю ручного режиму роботи, а також формування повторних запитів про сплату комісій, формування щомісячних, щоквартальних звітів, що характеризують кількість оброблених SWIFT-повідомлень і отриманий прибуток відділу.

Система призначена для використання в комерційному банку.

Програма chrgas.exe не може застосовуватися окремо від програмного комплексу Back_office і отже поза меж фінансової організації.

1.3. Підстава для розробки

Підставою для розробки кваліфікаційної роботи бакалавра на тему «Розробка інформаційної системи формування та контролю за проведенням повідомлень MT191 (запит про сплату комісій) комерційного банку» є наказ по Національному технічному університету «Дніпровська політехніка» від __.__. 2021р. № ____-_.

1.4. Постановка завдання

Метою кваліфікаційної роботи є створення інформаційної системи для автоматизації процесу збору статистичної інформації про роботу банку і на підставі цих даних створення повідомлень про сплату комісій третіх банків (MT191), а також автоматизація процесу формування звітів та надання необхідних документів для проведення розслідувань та інших операцій з цінними паперами

В рамках виконання роботи необхідно розробити автоматизовану систему для формування та контролю за проведенням повідомлень MT191 (запит про сплату комісій).

Для виконання роботи необхідно виконати наступні етапи проектування та розробки ІС:

1. Проаналізувати предметну галузь об’єкту розробки.

2. Обґрунтувати структуру, характер і організацію вхідних і вихідних даних.

3. Визначити функціональні характеристики розробленої системи.

4. Розробити алгоритм роботи системи, методи, відповідно до яких проводиться необхідний розрахунок та її логічну структуру.

5. Забезпечити можливість зв'язку програми з іншим ПЗ, яке впроваджене в банку.

6. Програмно реалізувати задану систему.

7. Протестувати систему та виправити можливі недоліки.

Програма chrgas.exe не може застосовуватися окремо від програмного комплексу Back_office і отже поза межами фінансової організації.

Введення інформації повинно здійснюватися за допомогою консолі з використанням діалогових віконних систем, доступ до яких вимагає аутентифікації користувача.

Вивід даних повинен бути реалізований в двох режимах:

– діалоговому, при якому на запити користувача система виводить на

екран інформацію, що цікавить користувача;

– файловому, при якому звіти різної спрямованості формуються в залежності від введених користувачем даних. Запити на формування звітів програма формує в діалоговому режимі.

1.5. Вимоги до програми або програмного виробу

1.5.1. Вимоги до функціональних характеристик

Інформаційна система chrgas призначена для формування повідомлень про сплату комісій третіх банків по кредитним повідомленнями (повідомлення типу MT103) та автоматизації процесу розслідувань про неоплачені комісії з можливістю ручного режиму роботи, а також формування повторних запитів про сплату комісій, формування щомісячних, щоквартальних звітів, що характеризують кількість оброблених SWIFT-повідомлень і отриманий прибуток відділу.

Система повинна надавати наступні функціональні можливості:

– формування повідомлень про сплату комісій третіх банків по кредитними повідомленнями (повідомлення типу MT103);

– автоматизація процесу розслідувань по несплаченим комісіям з можливістю ручного режиму роботи;

– формування повторних запитів про сплату комісій;

– формування щомісячних, щоквартальних звітів, які характеризують кількість оброблених повідомлень і отриманий прибуток відділу.

1.5.2. Вимоги до інформаційної безпеки

Безпека даних включає їх цілісність і захист. Цілісність даних - стійкість даних, що зберігаються до руйнування і знищення, пов'язаних з несправностями технічних засобів, системними помилками і помилковими діями користувачів.

Цілісність даних полягає у логічній і фізичній цілісності даних.

Логічна цілісність - виражається у вигляді обмежень або правил збереження даних, несуперечність яких не повинна порушуватися в базі. Можна вказати межі значень при описі структури бази даних або таблиць бази.

Фізична цілісність виражається в захисті даних від фізичного руйнування в результаті збоїв і відмов устаткування. Для цього використовують контрольні копії даних, тобто запам'ятовування станів бази даних у контрольних точках, а також ведення системного журналу, в якому відображаються усі маніпуляції над базою даних.

Одним з видів захисту бази даних являється різноманітність прав доступу, що гарантує доступ до бази даних тільки санкціонованих користувачів.

Введення інформації в базу даних повинно бути мінімізоване, тобто інформація вводиться однократно, а її зміни відображаються тільки у виведених даних.

База даних повинна забезпечувати швидкий і гнучкий доступ до даних.

1.5.3. Вимоги до складу та параметрів технічних засобів

При роботі з системою повинні дотримуватися умови, що забезпечують нормальний режим експлуатації IBM PC- сумісних машин. Кількість персоналу, що працює з програмним продуктом - одна людина. Працювати з програмою повинен висококваліфікований оператор баз даних.

Для технічних засобів повинна бути обрана конфігурація, що забезпечує цілодобову роботу програмних комплексів з резервуванням даних. Для цього повинно бути використано два сервера, один з яких виконує функцію прийому і відправки повідомлень за допомогою програмного продукту TurboSwift, а також здійснює всю пост обробку повідомлень:

– аналіз повідомлень і маршрутизацію прийнятих повідомлень;

– аналіз і перевірку на наявність помилок відправляються банком SWIFT - повідомлень;

Другий сервер виконує функцію резервування даних і є копією першого.

1.5.4. Вимоги до інформаційної та програмної сумісності

Розроблена система повинна взаємодіяти з банківською БД і автоматизувати процес обробки архівних повідомлень, які зберігаються в базі даних керованих СУБД SQL AnyWhere 17.0.

РОЗДІЛ 2
ПРОЄКТУВАННЯ ТА РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1. Функціональне призначення системи

Система автоматичної обробки комісій третіх банків розроблена з метою підвищення продуктивності праці персоналу відділу, шляхом автоматизації процесу обробки отриманої банком інформації використовуючи програмно - технічні засоби. Розроблена система забезпечує автоматизацію процесу аналізу отриманих банком повідомлень, формування статистичних даних і на підставі цих даних створює повідомлення про сплату комісій третіх банків MT191. Система відправляє сформовані повідомлення банкам - відправникам MT103, з яких запитується комісія. Крім цього, на підставі отриманих статистичних даних, система визначає банки-неплатники з метою формування і відправки повторних запитів про необхідність сплати комісій і забезпечує формування щомісячних і щоквартальних звітів, які відображають отриманий прибуток відділу.

Розробка даного продукту необхідна для швидкої і ефективної обробки повідомлень про кредитування MT103, формування повідомлень про сплату комісій третіх банків MT191, а також пошуку банків, які відмовляються сплачувати комісії, з метою розслідування і усунення причин таких відмов з повторним повідомленням банків про необхідність оплатити комісії.

Дана розробка підвищує надійність і швидкість обробки фінансових повідомлень, а також знижує навантаження на персонал відділу, шляхом автоматизації операторської роботи.

2.2. Опис застосованих математичних методів

В даній роботі математичні методи та моделі не використовуються.

2.3. Опис використаних технологій та мов програмування

Для реалізації заданої роботи було обрано візуальне середовище програмування Microsoft Visual Studio 2017 і сервери баз даних Sybase: SQL Anywhere 17.0, і SYBASE System 11.5 Adaptive Server. Вибір був зроблений виходячи з наявних на підприємстві програмних засобів. Для організації обміну даними між різними системами управління БД, використовуємо програмну модель доступу до БД ADO.NET.

Програма написана мовою C ++. Вона широко використовується для розробки програмного забезпечення, будучи одним з найпопулярніших мов програмування. Область його застосування включає створення операційних систем, різноманітних прикладних програм, драйверів пристроїв, додатків для вбудованих систем, високопродуктивних серверів, а також ігор. Існує безліч реалізацій мови C ++, як безкоштовних, так і комерційних і для різних платформ. Наприклад, на платформі x86 це GCC, Visual C ++, Intel C ++ Compiler, Embarcadero (Borland) C ++ Builder і інші. C ++ зробив величезний вплив на інші мови програмування, в першу чергу на Java і C #.

Мова виникла на початку 1980-х років, коли співробітник фірми Bell Labs Бйорн Страуструп придумав ряд удосконалень до мови C під власні потреби. Коли в кінці 1970-х років Страуструп почав працювати в Bell Labs над завданнями теорії черг (в додатку до моделювання телефонних викликів), він виявив, що спроби застосування існуючих в той час мов моделювання виявляються неефективними, а застосування високоефективних машинних мов занадто складно через їх обмеженою виразності.

Стандарт C ++ складається з двох основних частин: опис ядра мови і опис стандартної бібліотеки.

Розвиток мови супроводив розвиток крос-компілятора cfront. Нововведення в мові відбивалися в зміні номера версії крос-компілятора. Ці номери версій крос-компілятора розповсюджувалися і на саму мову, але стосовно до теперішнього часу мова про версії мови С++ не ведуть. Лише в 1998 році мова стала стандартизованим.

Стандартна бібліотека С++ включає стандартну бібліотеку Сі з невеликими змінами, які роблять її більш відповідною для мови С ++. Інша велика частина бібліотеки С++ заснована на Стандартній Бібліотеці Шаблонів (STL). Вона надає такі важливі інструменти, як контейнери (наприклад, вектори і списки) і ітератори (узагальнені вказівники), що надають доступ до цих контейнерів як до масивів. Крім того, STL дозволяє схожим чином працювати і з іншими типами контейнерів, наприклад, асоціативними списками, стеками, чергами.

Доступ до можливостей стандартної бібліотеки С++ забезпечується за допомогою включення в програму (за допомогою директиви #include) відповідних стандартних заголовків файлів. Всього в стандарті С++ 11 визначено 79 таких файлів. Засоби стандартної бібліотеки оголошуються які входять в простір імен std. Заголовки, імена яких відповідають шаблону «X», де X - ім'я заголовки стандартної бібліотеки С без розширення (cstdlib, cstring, cstdio та ін.). Містять оголошення, відповідні даної частини стандартної бібліотеки С. Стандартні функції бібліотеки С також знаходяться в просторі назв std.

С ++ містить засоби розробки програм контрольованої ефективності для широкого спектра задач, від низькорівневих утиліт і драйверів до вельми складних програмних комплексів. Зокрема висока сумісність з мовою Сі: код на Сі може бути з мінімальними переробками скомпільована компілятором С ++. Візуально-мовний інтерфейс є прозорим, так що бібліотеки на Сі можуть викликатися з С++ без додаткових витрат, і більш того при певних обмеженнях код на С ++ може експортуватися зовні не відрізнятися від коду на Сі.

До числа недоліків мови можна віднести:

– відсутність системи модулів C++ успадкував від Сі підключення заголовних файлів за допомогою препроцесора. Це змушує дублювати опису об'єктів, породжує неочевидні вимоги до коду і збільшує обсяг компілюємого тексту, а значить і час компіляції;

– успадковані від Сі небезпечні і провокують помилки можливості (макроозначення, адресна арифметика, неявне приведення типів, можливість прямого управління розподілом пам'яті).

– відсутність вбудованих механізмів статичної валідації часу життя об'єктів, що приводить до раптового краху програм через звернення до знищеної змінної, або через неправильну багатопотокової роботи з об'єктами.

– шаблони породжують об'ємний і не завжди оптимальний код. Часткове визначення шаблонів ускладнює як сама мова, так і програми, де воно використовується.

Єдиним прямим нащадком C ++ є мовою D, задуманий як переробка C ++ для усунення найбільш очевидних його проблем. Автори відмовилися від сумісності з Сі, зберігши синтаксис і багато базові принципи C ++ і ввівши в мову можливості, характерні для нових мов. У D немає препроцесора, заголовків файлів, множинного спадкоємства, але є система модулів, інтерфейси, асоціативні масиви, підтримка unicode в рядках, прибирання сміття (при збереженні можливості ручного управління пам'яттю) вбудована багатопоточність, висновок типів, явне оголошення чистих функцій і незмінних значень. Використання D вельми обмежена, вважати його реальним конкурентом C ++ можна.

Sybase SQL AnyWhere є повнофункціональну СУБД на Intel-платформі для мобільних і невеликих груп користувачів. Дана СУБД дозволяє розробляти програми на основі технології "клієнт-сервер" на платформах Windows NT, Windows 7, OS/2, NetWare, Solaris / Sparc, HP-UX, AIX, DOS і QNX.

СУБД SYBASE SQL Anywhere є складовим елементом системи SYBASE System 11 і її подальшої версії SYBASE System 11.5 Adaptive. Основним

елементом цих систем є потужна СУБД SYBASE SQL. Server. Вона дозволяє зберігати величезні обсяги інформації і обробляти запити до баз даних із застосуванням технології клієнт-сервер. СУБД SYBASE SQL Server і SYBASE SQL Anywhere взаємно доповнюють один одного. Розробники SYBASE SQL Anywhere намагалися досягти максимальної сумісності баз даних, що створюються цієї СУБД, з базами даних SYBASE SQL Server. Саме цим пояснюється включення в SQL Anywhere деяких елементів SYBASE SQL Server.

SYBASE SQL Anywhere може поставлятися як в мережевому варіанті СУБД, так і автономному варіанті. В останньому випадку всі компоненти СУБД функціонують на тому ж комп'ютері, що і додаток. Для додатків не має значення, який варіант СУБД вони використовують. В рамках локальних мереж взаємодію з додатками-клієнтами здійснюється за допомогою таких мережевих протоколів як NetBIOS, TCP/IP і IPX. Бази даних, створені засобами SQL Anywhere, сумісні з подальшими версіями і стерпні між платформами, на яких може функціонувати дана СУБД.

SYBASE SQL Anywhere має вбудований інтерфейс з рядом програмних систем, наприклад таких як Powersoft PowerBuilder, Optima C++ і Power Designer. Взаємодія з іншими системами і додатками здійснюється за допомогою наступних програмних інтерфейсів:

1. Специфікація ODBC. Дана специфікація надає користувачам уніфікований інтерфейс між додатками і реляційними базами даних, в тому числі і базами даних SQL. Цей інтерфейс являє собою сукупність функцій. Ці функції організовують взаємодію додатків з базами даних в операційних системах Windows 3.x, Windows 2003, OS/2 і Windows NT. Специфікація ODBC представляє собою інтерфейс нижнього рівня.

2. Вбудований SQL (Embedded SQL interface) дозволяє "вбудовувати" SQL-оператори прямо в тексти програм-додатків на мовах С або С ++. Згодом такі додатки піддаються обробці спеціальним препроцесором, який замінює SQL-оператори на виклики відповідних функцій і процедур. Після цього

перетворені тексти програм представляють собою тексти програм на мові програмування С ++.

3. DDE-інтерфейс. Це технологія динамічного зв'язування об'єктів використовується в операційній системі Windows 3.x .. Для застосування цієї технології потрібне використання відповідних програмних засобів, що підтримують функціонування DDE-клієнта. З іншого боку, в якості серверів необхідно використовувати додатки, що забезпечують режим DDE-сервера. До числа останніх відносяться такі програмні додатки, як Microsoft Access, Microsoft Excel і ряд інших. В СУБД SYBASE SQL Anywhere роль DDE-сервера виконує WSQL (Watcom SQL) DDE Server.

4. Високорівневий власний інтерфейс WSQL HLI (Watcom SQL High-level interface) забезпечує на "високому" рівні взаємодію між SQL Anywhere і додатками в середовищах операційних систем Windows 3.x, Windows 7, OS/2 і Windows NT. Даний інтерфейс реалізований для систем програмування аналогічних Visual Basic (для Windows 7 і Windows NT) і REXX (для OS/2).

Для реалізації даної роботи, найкраще підходить специфікація ODBC, що надає уніфікований інтерфейс між додатками і реляційними базами даних.

СУБД SYBASE SQL Anywhere має в своєму складі мережевий і автономний варіанти СУБД. Ці варіанти реалізовані у вигляді компонент SQL Anywhere server/client і SQL Anywhere engine. SQL Anywhere server/client представляє мережевий варіант СУБД. Він організовує взаємодією додатків з базами даних в рамках локальної обчислювальної мережі за технологією "клієнт-сервер".

Крім локальної мережі, СУБД SYBASE SQL Anywhere надає можливість використовувати технологію "клієнт-сервер" і в рамках мережі Internet. Для цього необхідно, щоб на комп'ютері з сервером бази даних був організований веб-сервер, і до складу його програмного забезпечення входив додаток SYBASE Netlmpact Dynamo.

Мережевий варіант включає в себе віддалений сервер і клієнтську компоненту; SQL Anywhere server і SQL Anywhere client, відповідно.

Функціонування СУБД в рамках мережі полягає в наступному. На одній ЕОМ, що грає роль сервера бази даних, запускається віддалений сервер - SQL Anywhere server. Його завдання полягає в очікуванні запитів від клієнтських ЕОМ, їх обробці та відправленні результатів клієнтів. Під час роботи сервера на клієнтських ЕОМ функціонує компонента SQL Anywhere client. Вона приймає від клієнтських додатків запити. Ці запити являють собою оператори мови SQL, наприклад, SELECT (вибрати дані), UPDATE (модифікувати дані), CALL (викликати збережену процедуру) та інші. Отримані запити компонента SQL Anywhere client відправляє по мережі сервера бази даних і очікує повернення від нього результатів цих запитів.

Компонента SQL Anywhere engine є автономним варіантом даної СУБД і являє собою локальний сервер баз даних. Він реалізує технологію "клієнт-сервер" в умовах, коли сервер і клієнти функціонує на одній ЕОМ. База даних зберігається на цій же ЕОМ. Сервер приймає запити безпосередньо від додатків, обробляє їх і відправляє додаткам результати запитів.

Клієнтські програми не розрізняють, з яким варіантом СУБД вони працюють. Одні і ті ж запити виконуються для користувача абсолютно однаково за одними і тими ж правилами.

Компоненти SQL Anywhere server/client і SQL Anywhere engine складають основу СУБД. Однак поряд з ними до складу СУБД входить і ряд інших компонентів. Всі разом ці компоненти складають комплект поставки SQL Anywhere, в який входять:

– SQL Anywhere server / client - мережевий варіант СУБД;

– SQL Anywhere engine - автономний варіант СУБД;

– SQL Remote - компонента підтримки режиму реплікації;

– Sybase SQL Central - утиліта адміністрування баз даних;

– ISQL (Interactive SQL) - утиліта інтерактивної взаємодії з базами даних;

– набір додаткових утиліт для створення баз даних, їх стиснення, знищення і т.д.

Наведені компоненти є елементами стандартної поставки. Крім стандартної поставки СУБД може пропонуватися у вигляді професійної поставки. Вона має ще більше число складових.

Бази даних, що створюються СУБД SYBASE SQL Anywhere, являють собою реляційні бази даних. Ці бази даних складаються з сукупності об'єктів. Такими об'єктами є:

– Таблиці даних - зберігають дані, що становлять основний зміст БД;

– Ключі - сукупності атрибутів, що утворюють ключі (первинні і зовнішні), призначені для здійснення прискореного пошуку даних і забезпечення обмежень посилальної цілісності;

– Індекси - спеціальні таблиці, призначені для швидкого пошуку необхідної інформації в таблицях даних;

– Представлення (Views) - пов'язані сукупності підмножин таблиць даних, що надаються користувачам для обмеження їх доступу до таблиць даних. При цьому, до одних таблиць у доступі повністю, а в інших таблицях доступ дозволяється тільки до деяких записів цих таблиць;

– Збережені процедури і функції: збережені в базі даних підпрограми на мові SQL, скористатися які може будь-який користувач, який має на це право;

– Тригери - підпрограми, що активізуються при настанні певних подій, наприклад, видалення запису з таблиці, модифікація записів і т.д. Тригери є потужним засобом забезпечення цілісності даних;

– Користувальницькі типи даних - типи даних, створювані користувачем на підставі базових типів даних СУБД;

– Системні таблиці - зберігають всю інформацію про схему бази даних і що містяться в ній об'єктах.

У даній СУБД використовується діалект мови SQL - Watcom SQL. Він відповідає стандартам ANSI SQL/89 Level 2 і IBM SAA. Крім того, використовуваний діалект SQL підтримує нові можливості і розширення стандартів ANSI SQL/92 і IBMs DB2

Вся інформація бази даних може розміщується в декількох областях.

Областю є файл з розширенням db, в якій зберігається вся база даних або один з її фрагментів. Кожна область характеризується своїм ім'ям і файлом, відповідним цій галузі. Спочатку база даних займає тільки одну область з ім'ям SYSTEM, якій відповідає базовий файл (root file). Потім, у міру необхідності, простір зовнішньої пам'яті бази даний може розширюватися за рахунок додавання нових областей. Ці файли областей можуть бути розміщені в будь-якому каталозі на будь-якому диску і будь-якому вузлі локальної мережі.

Фізично кожен файл (область) складається зі сторінок фіксованого обсягу. Розмір сторінок встановлюється при створенні бази даних. У сторінках розміщуються об'єкти бази даних. Залежно від розміру об'єктів кожна сторінка може містити кілька примірників об'єктів або фрагмент одного з об'єктів.

Поряд з файлами областями до складу бази даних входить файл для зберігання журналу змін БД - файл з розширенням log. Наявність журналу змін є основою маніпулювання даними в даній СУБД з використанням механізму транзакцій. Такий підхід забезпечує високий захист баз даних від виникаючих програмних і апаратних збоїв і відмов.

Для реалізації бази даних, в якій буде зберігатися вся необхідна для роботи програми інформація, в даній роботі використана технологія роботи з БД на основі SQL-AnyWhere 17.0.

Для реалізації взаємозв'язку з БД банку використана технологія доступу до баз даних ADO.NET, використовуючи при цьому специфікацію мови SQL - Watcom.

Для реалізації клієнтської частини інформаційної системи використана середу візуального програмування Microsoft Visual 2010.

2.4. Опис структури системи та алгоритмів її функціонування

2.4.1. Розробка БД системи

2.4.1.1. Опис логічної структури бази даних

Логічна структура бази даних наведена на рис. 2.1.

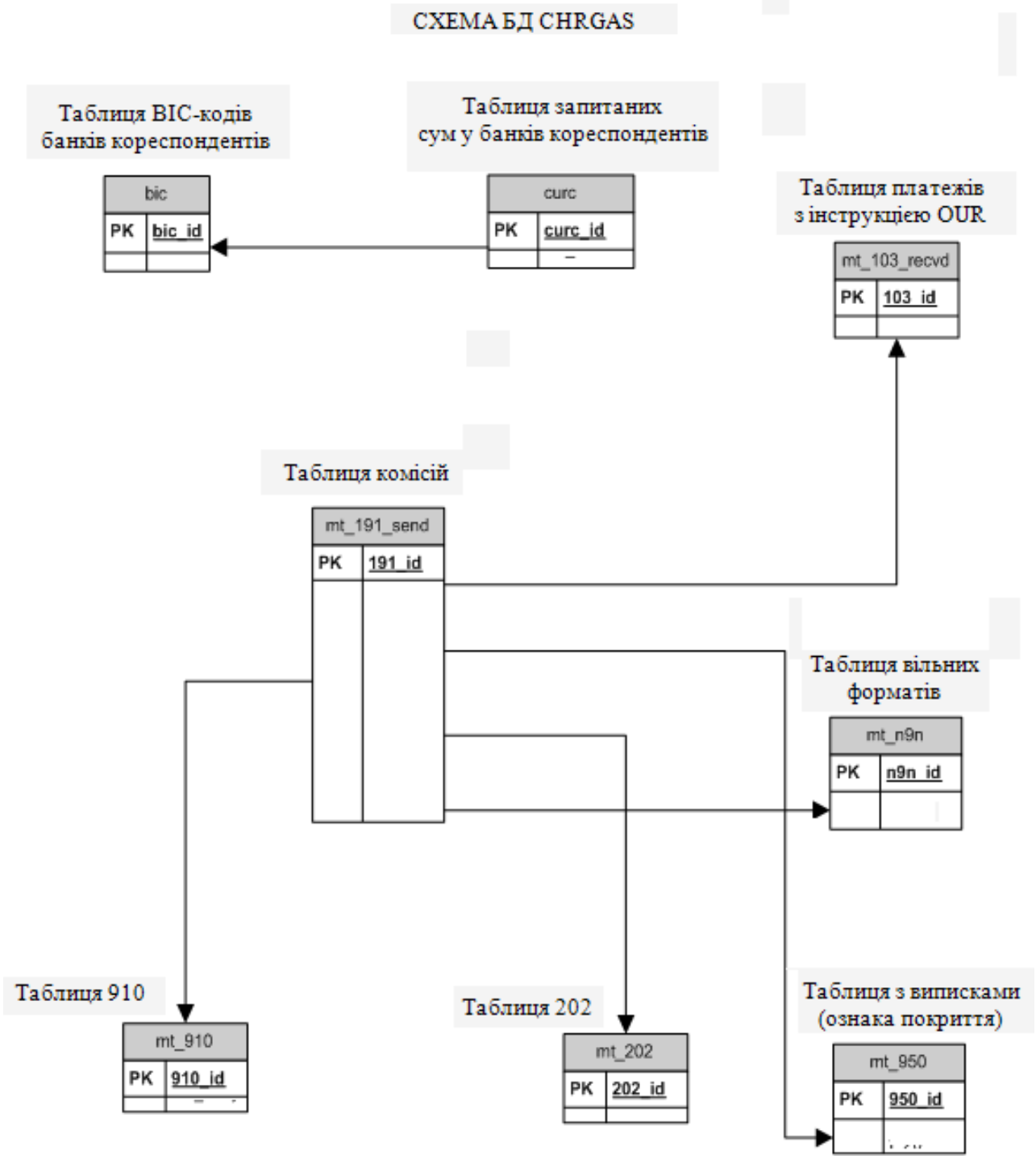

Рис. 2.1. Логічна структура бази даних

У даній логічної моделі приведений зв'язок оброблюваних програмою chrgas.exe даних. Основними даними є повідомлення, що зберігаються в таблиці mt_191_sent, саме для їх обробки і створювалося дане ПЗ. Таблиці bic_ і curc_ використовуються для визначення BIC- кодів банків і валют, на які відділ може відсилати повідомлення MT191.

Всі інші таблиці в БД chrgas.db використовуються для зберігання даних про рух по рахунках при сплаті банками запитаних комісій. Дана інформація використовується програмою для автоматизації процесу розслідування неоплачених комісій. У процесі формування MT191, всі необхідні дані програма chrg.as.exe отримує з БД back_off.db. У даній БД повідомлення зберігаються в двох пов'язаних між собою таблицях: mx_ і ms_.

Призначення таблиць БД chrgas.db:

1. У таблиці mt_103_recvd зберігаються посилальні дані, необхідні для швидкого пошуку потрібної інформації. В цій таблиці зберігається і текст оригінального повідомлення MT103. Розглянемо ці довідкові дані:

– референс повідомлення, 20 поле;
– дата валютування;
– валюта;
– сума комісій;
– відправник повідомлення;
– поле 72, в якому вказана інформація відправника одержувачу (ПриватБанк в цьому полі вказує ідентифікатор клієнта-замовниками MT103);
– текст повідомлення MT103.

2. Таблиця mt_910 не містить повний текст повідомлення, проте містить наступну інформацію:

– посилання на відправлене MT191, 21 поле;
– референс повідомлення, 20 поле;
– дата валютування;
– валюта;

– сума комісій;

– відправник повідомлення.

3. Таблиця mt_202 містить той же набір даних, що і таблиця mt_910.

4. Таблиця mt_950 містить референс повідомлення, поле 61 містить рядок, що описує рух по рахунку.

5. Таблиця mt_n9n містить:

– посилання на відправлене MT191, 21 поле;

– рефференс повідомлення, 20 поле;

– текст повідомлення.

2.4.1.2. Опис фізичної структури бази даних

Фізична модель бази описує фізичну інтерпретацію логічного її структури і являє собою таблиці бази даних, пов'язані між собою реляційними зв'язками. Кількість полів та їх типи наведені в схемі та відповідають фізичної реалізації БД. Відносини між таблицями показані на схемі у вигляді спрямованих стрілок.

Фізична структура БД приведена на рис. 2.2.

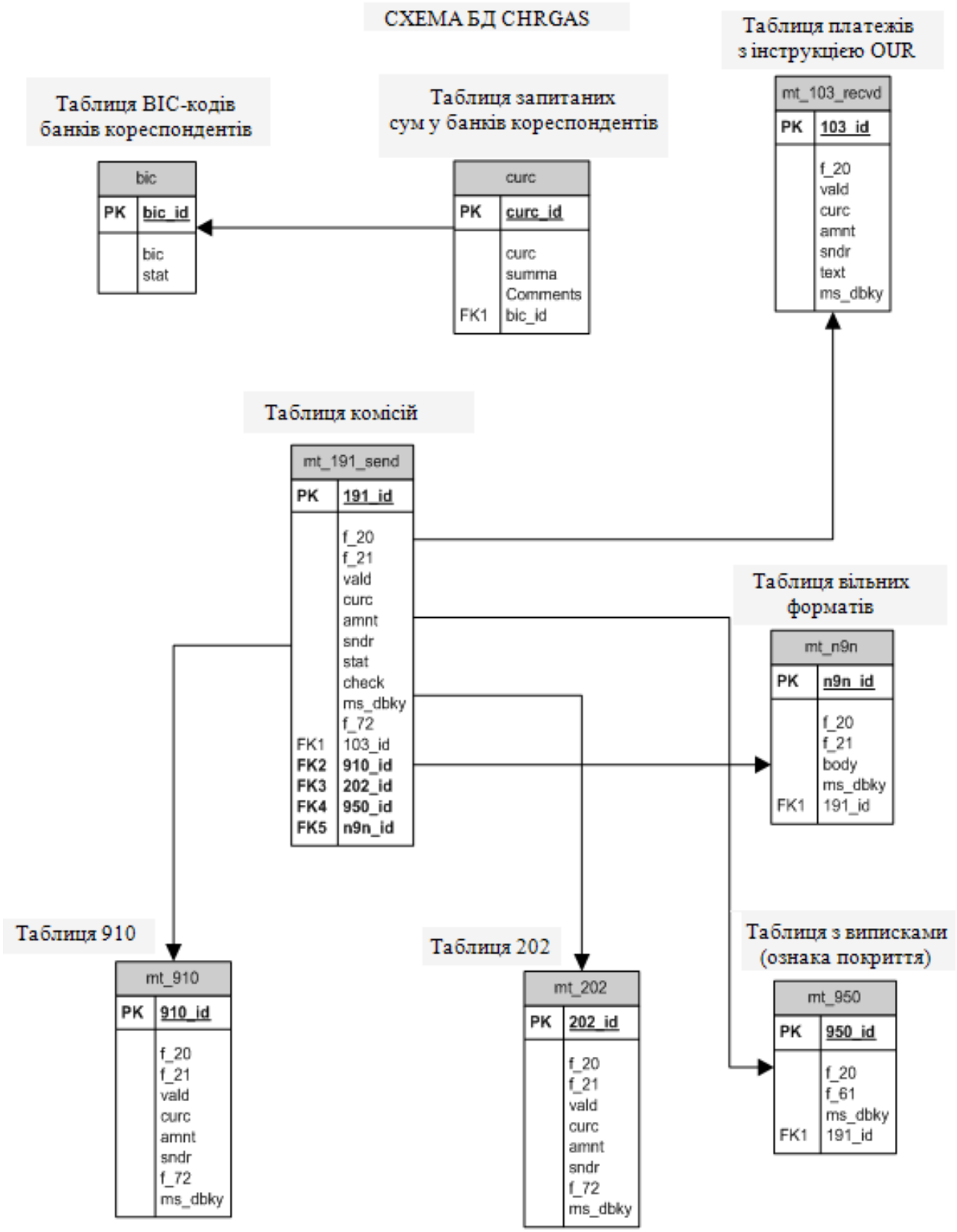

Рис. 2.2. Фізична структура бази даних

Наведемо типи даних, визначені в БД chrgas.db:

– tamnt, дані цього типу служать для представлення інформації про суму платежу та суму запитуваних комісій, тип даних реалізований від базового char с розмірністю 17 байт;

– tcurc, в змінних цього типу зберігається валюта, тип даних char (3);

– tid, змінна - генератор унікальних значень цілого типу, використовується для формування первинного ключа в БД, базовий тип-integer;

– ms_dbky, в змінній зберігається посилання з бази даних back_office повідомлення, наприклад поле 20 і 21, базовий тип char;

– tsndr, дані цього типу служать для представлення інформації про відправника МТ, базовий тип char;

– tvald, - дата валютування, базовий тип char.

2.4.2. Опис алгоритму і методики формування повідомлення MT191

Розроблений алгоритм, складається з двох частин:

– алгоритм формування повідомлення про сплату комісій;

– алгоритм проведення розслідувань.

Алгоритм формування повідомлення про сплату комісій полягає в знаходженні необхідних повідомлень МТ103 в базі даних back_off.db програми Back_office з метою формування повідомлень про оплату комісій, відсотків або інших витрат, про які раніше не було відомо одержувачу.

Знаходимо в back_off.db повідомлення із зазначеного діапазону дат (наприклад, можна вказати певну дату у вигляді 20210311 або із зазначенням діапазону дат: 20210311 20210320), і всі записи, що задовольняють таким умовам:

1. Необхідно, що б в поле: 71A: було вказано ключове слово OUR. Поле 71A вказує сторону, за рахунок якої оплачуються витрати перекладу платежу. В поле міститься одне з трьох кодових слів:

– OUR - всі витрати за рахунок відправника. При його наявності заборонено використовувати поле 71F, а поле 71G є необов'язковим;

– SHA - всі витрати на стороні відправника, повідомлення несуться платником, на стороні одержувача повідомлення одержувачем платежу. При його наявності заборонено використовувати поле 71G, а поле 71F необов'язково;

– BEN - всі витрати за рахунок одержувача. При його наявності обов'язкова присутність поля 71F і заборонено використовувати поле 71G.

Відділ SWIFT КБ «ПриватБанк», формує повідомлення MT191 тільки для тих поодиноких клієнтських переказів, у яких в поле: 71A: присутня опція OUR.

2. Проводимо перевірку BIC - відправника MT103, для визначення банків, на які Приватбанк не відсилає MT191.

3. Якщо BIC код присутній в списку заборонених банків, то обробка даного повідомлення MT103 в автоматичному режимі заборонена.

4. Проводимо перевірку суми і валюти відправника MT103 для визначення суми і валюти запитуваних комісій.

5. Формуємо MT191 з MT103 за наступною схемою:

– банк Banca Commerciale Italiana, Los Angeles, отримав від Niederoesterreichische Laendesbank-Hypotekenbank, Vienna, повідомлення клієнтського перекладу, за умовами якого йому доручалося сповістити бенефіціара про надходження коштів по телефону;

– Banca Commerciale Italiana просить Niederoesterreichische Laendesbank-Hypotekenbank оплатити витрати за цю телефонну розмову і перевести кошти на його рахунок в Chase Manhattan Bank, New York.

Рух інформації представлено на рис. 2.3. В табл. 2.1. наведено зміст та опис даного повідомлення.

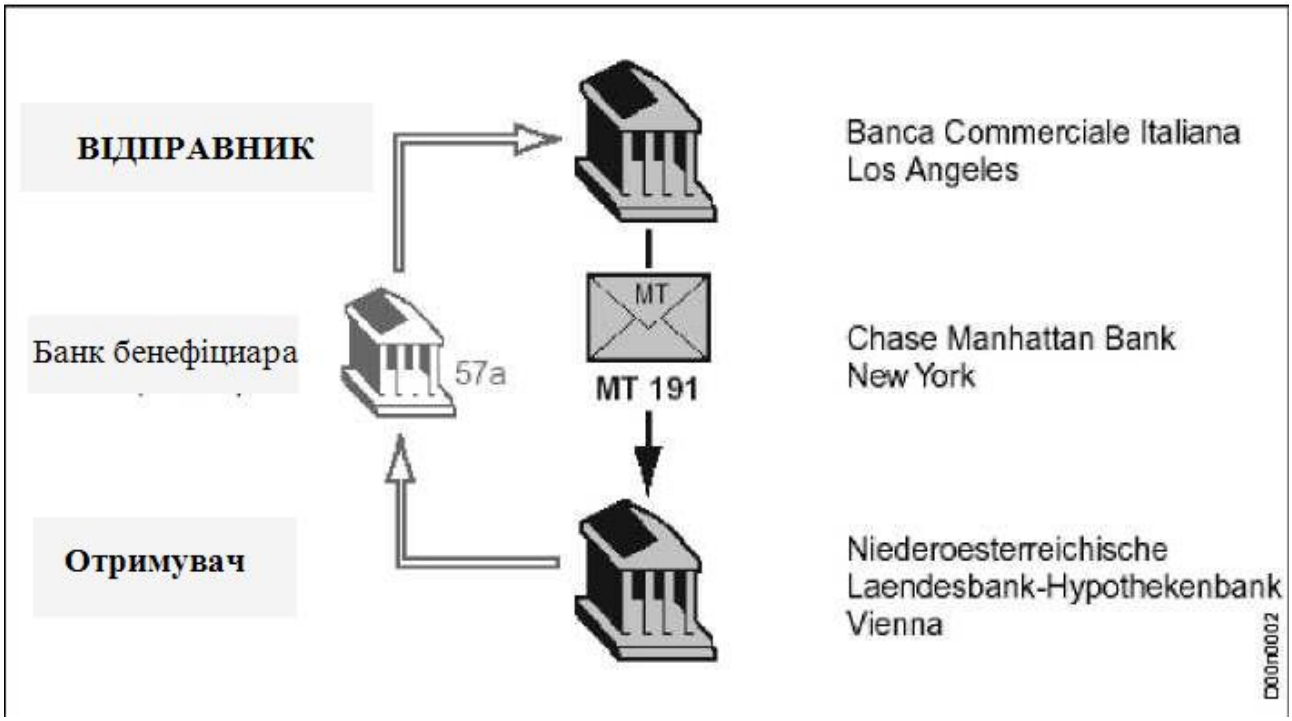

Рис. 2.3. Рух інформації повідомлень MT191 з MT103

Таблиця 2.1

Опис змісту повідомлення

Опис	**Формат**
відправник	BCITUS66
Тип повідомлення	191
одержувач	HYPNATWW
Текст повідомлення	
Референс операції:	20: 948LA
Референс вихідного платіжного доручення:	21: 516 722 / DEV
Код валюти / сума витрат:	32B: USD10,
Банк бенефіциара (1)	: 57A: CHASUS33
Обґрунтування витрат:	71B: / PHON /
Кінець тексту повідомлення / трейлер	

Алгоритм проведення розслідувань (інвестицій) полягає, в першу чергу, в автоматизації процесу перевірки оплати запитаних банком комісій. Комісія вважається оплаченою, якщо банк-відправник отримав:

– повідомлення MT202 з повідомленням про переміщення коштів на його рахунок.

– кредитне авізо MT910 з повідомленням про зарахування запитуваних комісій на рахунок.

Завершальним етапом є отримання банком-відправником MT191 виписки, яка повідомляє, що проводка за вказаним рахунком проведена.

Повідомлення MT202 і MT910 є взаємозамінним і тому, банку-відправнику MT191, досить отримати одне з них. Банк-платник для повідомлення про виплату комісій може надіслати повідомлення MTn99, яке не є обов'язковим.

Алгоритм проведення розслідувань, що розроблений в програмі наступний:

1. Знаходимо в базі даних chrgas.db кількість посилань на інформацію про відправлені MT191 за вказаний звітний період.

2. Формуємо рядок пошуку в базі chrgas.db у вигляді: 20 поле MT191_20 поле MT103_дата валютування_валюта_сума_поле 72_статус. Вся ця інформація використовується для пошуку в БД back_off.db Back_office необхідних повідомлень в зазначеному ланцюжку: MT103-> MT191-> MT910 (MT202) -> MT950 (940). За стандартами, в кожному повідомленні міститься посилання на пов'язане з ним, яке прийшло раніше MT. Однак, багато фінансових організацій неправильно або некоректно заповнюють дане поле 20 в MT, що ускладнює автоматизацію процесу обробки і розслідувань комісій третіх банків. Саме тому необхідно формувати рядок пошуку, що містить стільки надлишкової інформації.

3. По внутрішньому алгоритму контекстного перебору, знаходимо дані в базі chrgas.db, відповідні рядку пошуку.

4. Знайдену інформацію заносимо в БД chrgas.db.

5. Якщо ми досягли кінця ланцюжка пошуку MT103-> MT191-> MT910 (MT202) -> MT950 (940), то в БД chrgas.db, в поле status таблицю MT191, заносимо символ P, який говорить про те, що по даному запиту комісія

сплачена.

2.4.3. Опис логічної структури системи

Розроблена система обробляє інформацію, взаємодіючи при цьому з різними програмними модулями і системами. Для того, щоб здійснити аналіз даних, програма взаємодіє з базою даних back_off.db програмного комплексу Back_office. Для проведення статистичного аналізу з подальшою обробкою отриманих даних, програма chrgas.exe взаємодіє з БД chrgas.db, в якій зберігаються копії оброблених даних, отриманих з БД back_off.db і результати проведеного аналізу. Також в базі даних charges.db, зберігаються всі оброблені дані програмою garges.exe.

Система взаємодіє з БД Back_off за допомогою технології ADO.Net корпорації Microsoft, при цьому розробник проектував структуру ПЗ, використовуючи ідеологію об'єктно-орієнтованого програмування. Для роботи з базами даних chrgas.db і back_off.db в розробленому ПЗ для кожної таблиці був оголошений свій клас, що містить як змінні для роботи з базою даних, так і методи роботи з нею. Структура програми приведена на рис. 2.4.

Логічну схему взаємодії програм представлено на рис. 2.5.

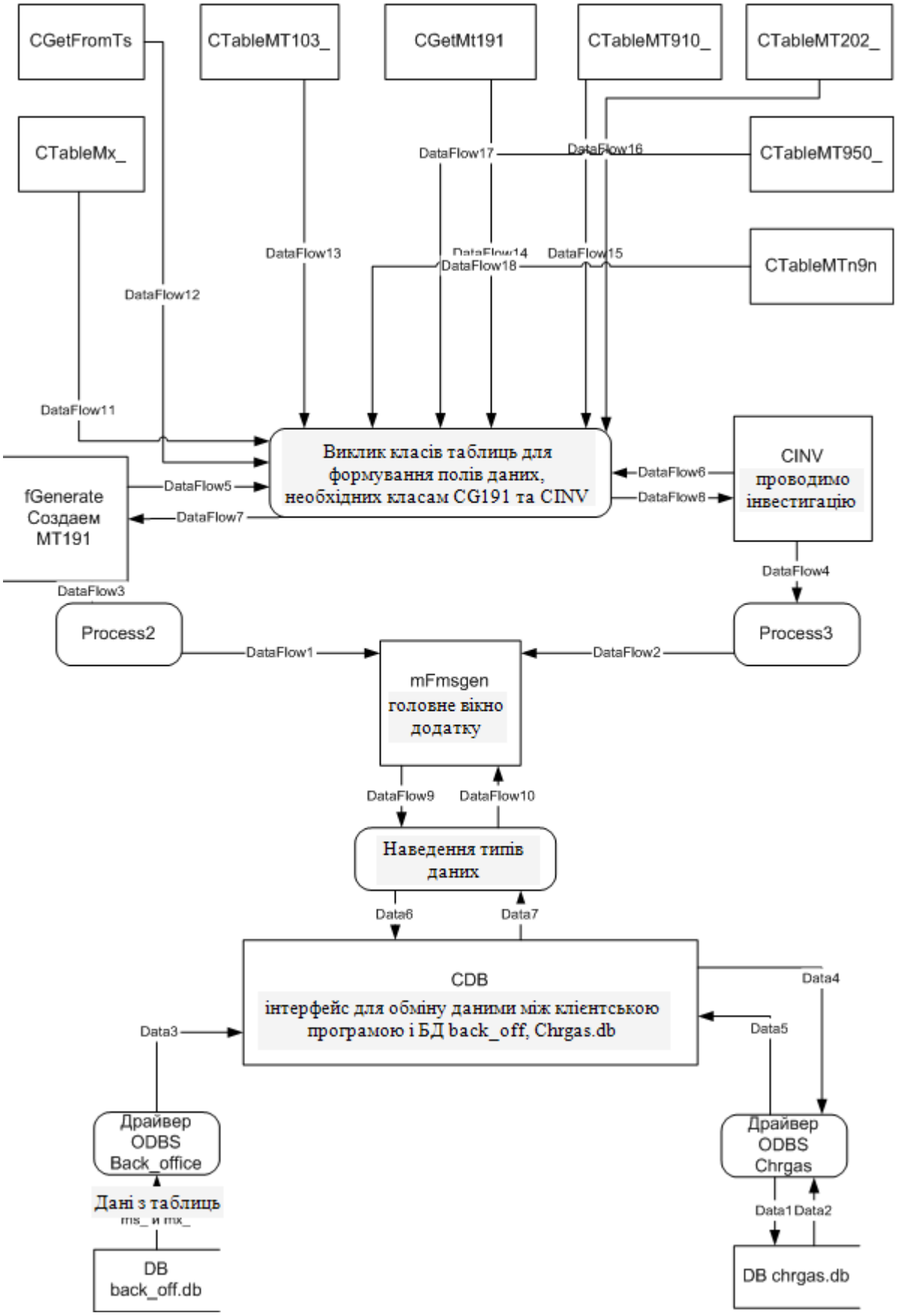

Рис. 2.4. Структура роботи програми.

На схемі відображені всі модулі, що використовує ПЗ даної системи. Загальні прийняті позначення в системі:

– DataFlow - інформаційний потік;

– SQL Anywhere server і SQL Anywhere client - віддалений сервер і клієнтська компонента, відповідно;

– Microsoft ODBC – специфікація, являє собою інтерфейс нижнього рівня для взаємодії додатків Chrgas.exe с БД back_off.db і chrgas.db;

– BDE Egine - являє собою інтерфейс доступу до БД, розроблений фірмою Borland і застосований в даній системі;

– високорівневий інтерфейс WSQL HLI (Watcom SQL High-level interface), який забезпечує на "високому" рівні взаємодію між SQL Anywhere і додатком TurboSwift в середовищах операційних систем Windows 7.x, OS / 2.

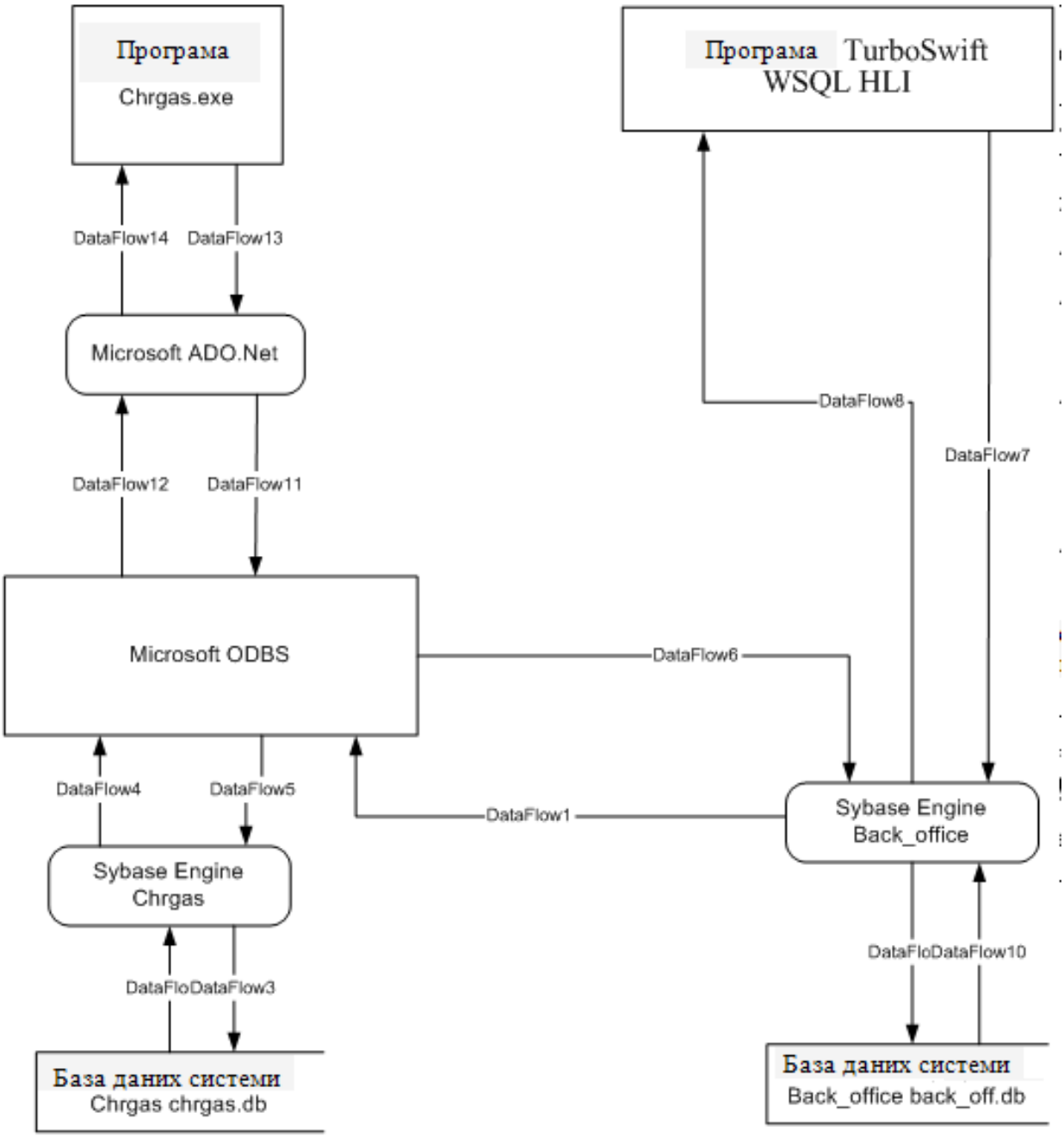

Рис. 2.5. Схема зв'язків системи з іншими програмами

2.5. Обґрунтування та організація вхідних та вихідних даних програми

Для автоматизації процесу формування комісій, вхідними даними є набір даних БД back_off.db програми Back_office, отриманих за допомогою виконання внутрішнього SQL запиту.

При проведенні розслідувань в ручному режимі, вхідними даними є дані,

внесені користувачем у відповідні поля діалогу.

Вихідні дані представлені в двох видах:

– електронному у вигляді текстових файлів і звітів про проведену роботу;

– діалоговому при ручному режимі проведення розслідувань.

2.6. Опис роботи розробленої системи

2.6.1. Використані технічні засоби

Для технічних засобів обрана конфігурація, що забезпечує цілодобову роботу програмних комплексів з резервуванням даних. Для цих засобів були обрані два сервера Dell PowerEdge 2600 Server, один з яких виконує функцію прийому і відправки повідомлень за допомогою програмного продукту Back_office, а також здійснює всю постобробку повідомлень:

– аналіз повідомлень і маршрутизацію прийнятих повідомлень;

– аналіз і перевірку на наявність помилок відправляються банком повідомлень.

Другий сервер виконує функцію резервування даних і є дзеркальною копією першого.

2.6.2. Використані програмні засоби

Розроблена система працює з декількома програмними продуктами сторонніх розробників, користуючись їх ресурсами. Програма працює з локальною базою даних chrgas.db під управлінням ОС Windows і взаємодіє за допомогою СУБД Sybase з базою даних back_off.db, програми Back_office. Програма Back_office і розроблена система працюють на одній робочій станції. Цей режим роботи системи був створений на вимогу замовника. Однак, система може працювати на будь-якому комп'ютері в мережі, на якому встановлено сервер БД Sybase і налаштований ODBC драйвер бази даних back_off.db.

2.6.3. Виклик та завантаження програми

Запуск програми здійснюється або через меню «Пуск», «Програми» операційної системи, або шляхом активації .exe файлу програми.

2.6.4. Опис інтерфейсу користувача

Метою створення даної системи є автоматизація процесу обробки повідомлень MT191 - запиту про сплату комісій та проведення звірки за кількістю заплачених комісій.

Для доступу до програми необхідно пройти процедуру аутентифікації. Всі необхідні відомості, для перевірки введених даних, програма отримує з БД chrgas.db. Процедура аутентифікації представлена на рис. 2.6.

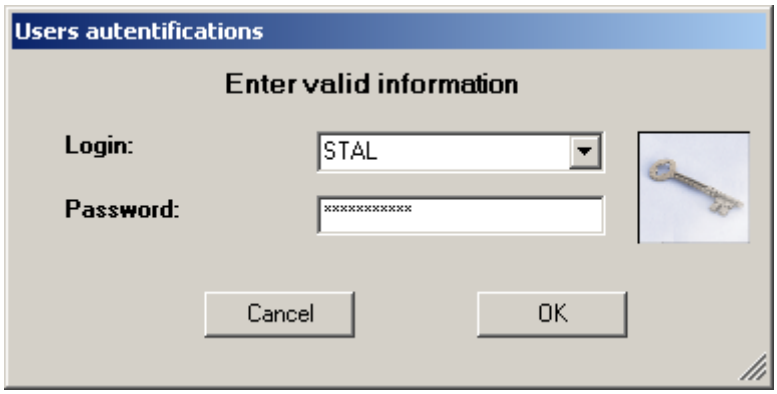

Рис. 2.6. Діалогове вікно аутентифікації

Після успішного проходження процедури аутентифікації користувачеві доступно головне вікно програми (рис. 2.7):

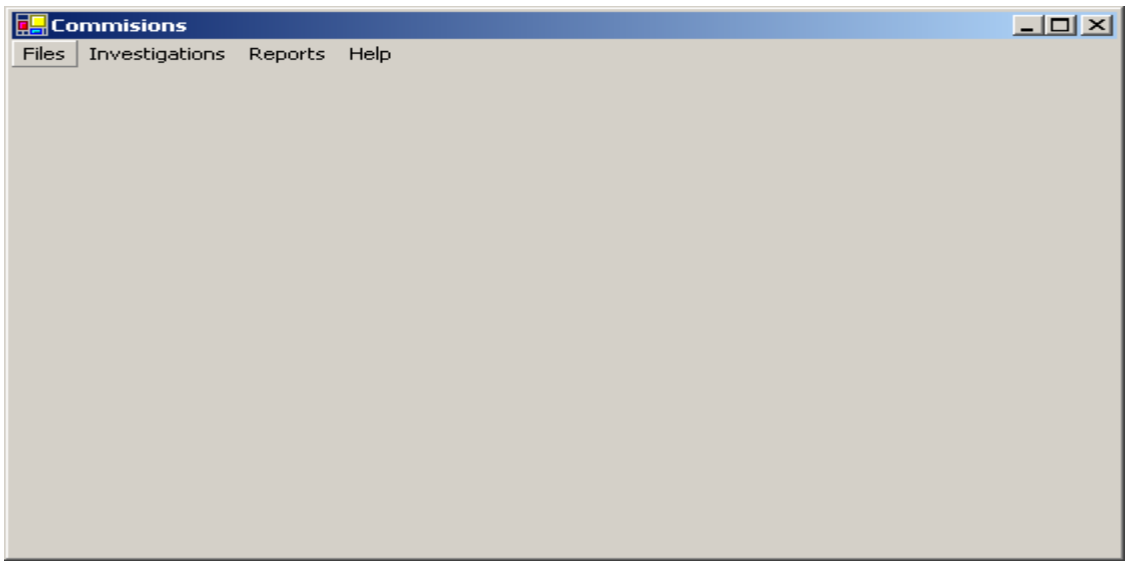

Рис. 2.7. Головне вікно програми

Для створення повідомлень і подальшої їх відправки в банки необхідно вибрати пункт головного меню File-> Generate Request (рис. 2.8).

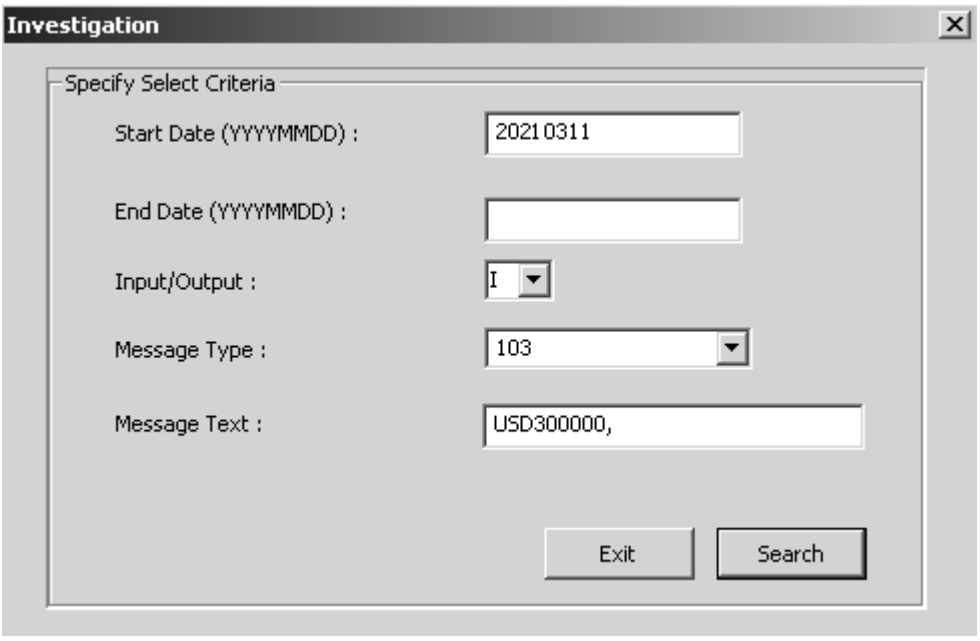

Рис. 2.8. Діалогове вікно пошуку SWIFT - повідомлення.

Для формування комісій необхідно ввести дату (або діапазон дат), для яких будуть проводитися пошуки платежів MT103 в БД Back_off.db, далі вхідними даними для розробленого ПЗ, буде набір полів, отриманих з бази

Back_office.

Для проведення розслідувань, вхідними будуть дані, введені користувачем в поля, відповідної форми.

Призначення полів форми:

– Start Date - в разі пошуку в діапазоні дат, змінна містить початкову дату для пошуку повідомлення, в іншому - містить дату операційного дня банку для пошук повідомлень;

– End Date - застосовується тільки для пошуку повідомлень в деякому діапазоні дат;

– Input / Output - змінна зберігає значення спрямованості повідомлення, по відношенню до КБ «ПриватБанк», вхідні/вихідні;

– Message Type - тип шуканих повідомлень;

– Message Text - текст в повідомленні, при наявності в даному полі будь-яких даних проводиться пошук повідомлення, яке містить введений користувачем текст.

Для формування звітів необхідно в меню головного діалогу вибрати пункт Reports-> Generate, після цього на екрані відобразитися діалогове вікно, представлене на рис. 2.9.

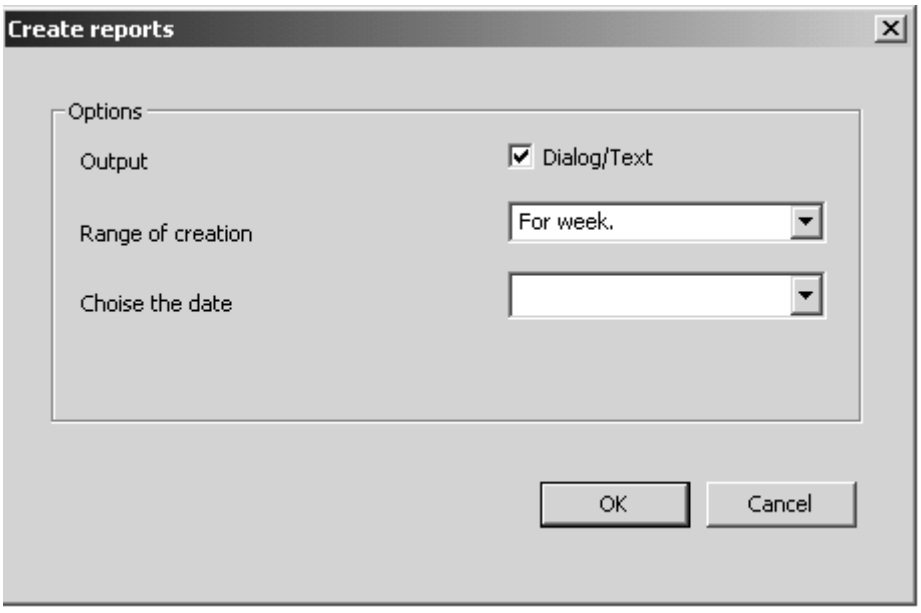

Рис. 2.9. Діалогове вікно створення звітів.

В даній формі, користувач вибирає метод виведення звіту та діапазон, для якого створюються звіти, і початкову дату діапазону. Наведемо опис полів форми:

– Output - визначає напрямок виведення сформованого звіту: на екран / в текстовий файл;

– Range of creation - задає діапазон створення звітів;

– Choice the creation - містить початкову дату діапазону.

Для здійснення адміністрування БД chrgas.db необхідно викликати File-> BD administrator. У діалоговому вікні можна встановлювати банки кореспондентів, а також суми і бізнес правила взяття комісій. У діалоговому вікні можна переглядати списки банків, з якими заборонено працювати по комісіям, а також банки - кореспонденти і пов'язані з ними бізнес правила, що визначають маршрути проходження платежів (рис. 2.10).

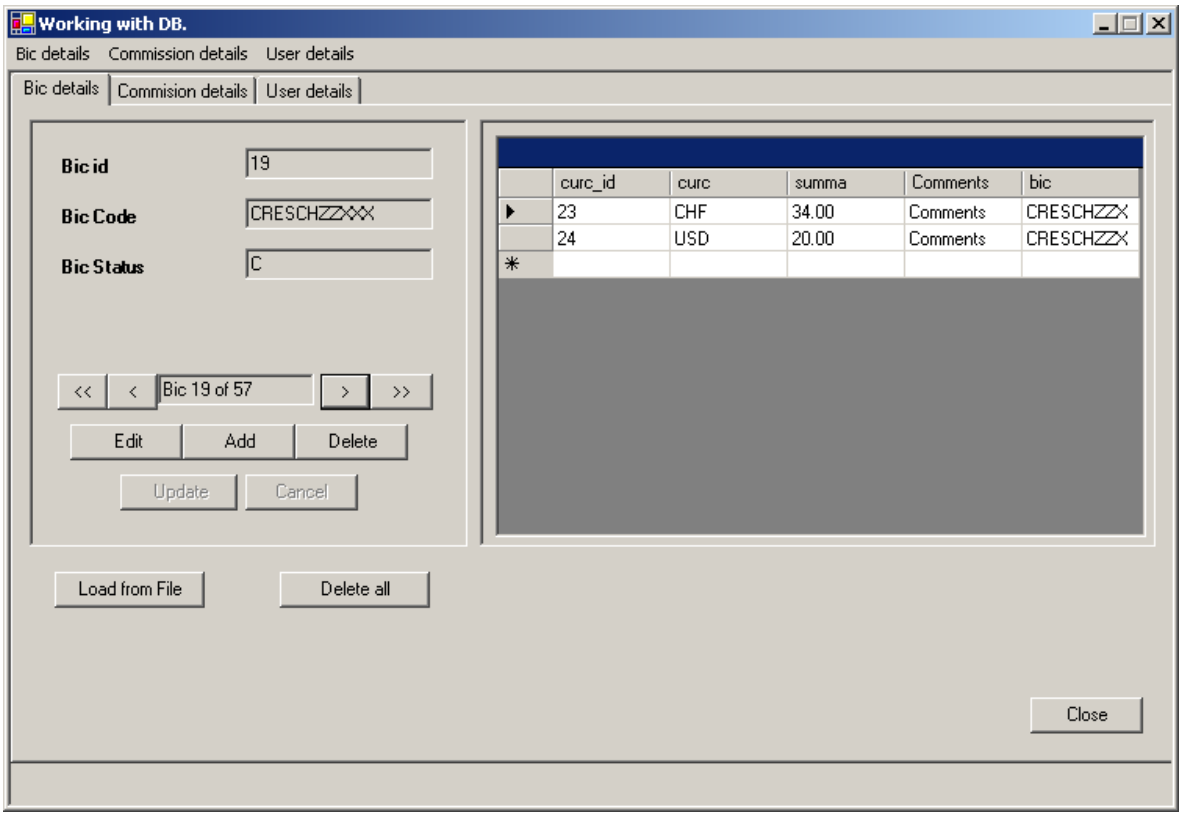

Рис. 2.10. Діалогове вікно банків-кореспондентів

Під бізнес - моделлю мають на увазі правила складання маршрутів

платежів із зазначенням банків - кореспондентів по валютах, а також сумою необхідної комісії. Дані моделі використовуються програмою в алгоритмах формування комісій (рис. 2.11).

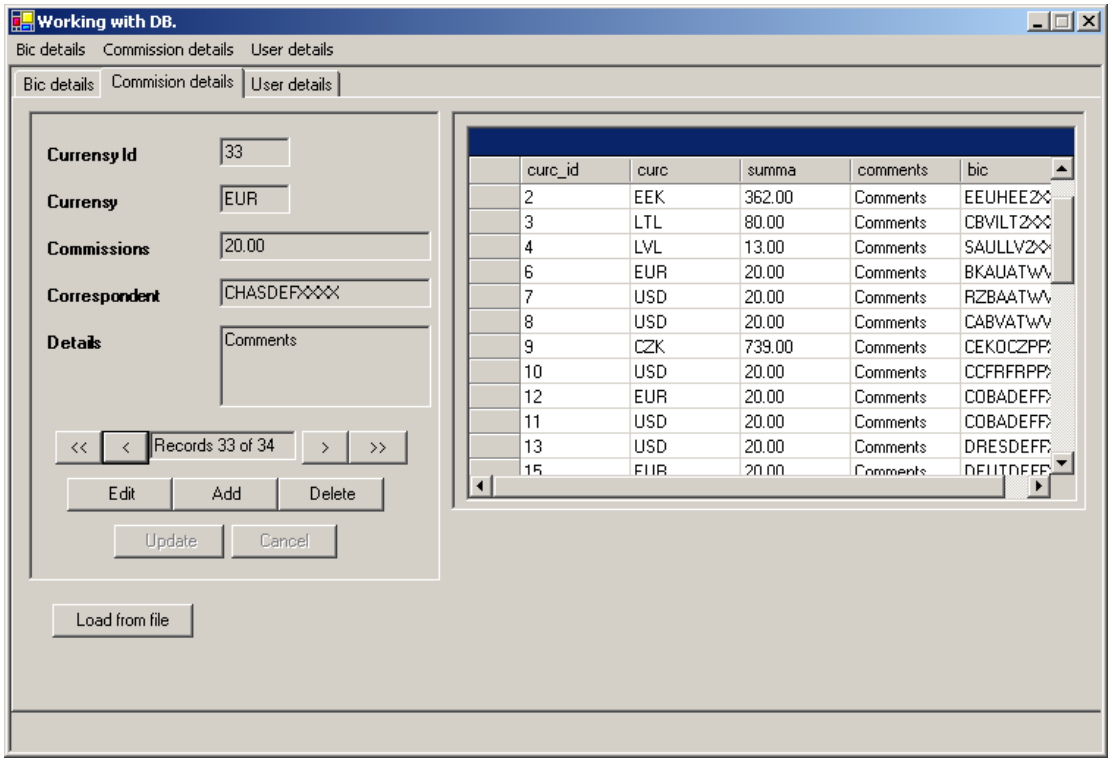

Рис. 2.11. Діалогове вікно для контролювання бізнес - моделей

Результати роботи програми поділяються на дві групи:

– в файловому вигляді;

– в електронному, з виведенням на екран в діалогове вікно.

У файловому вигляді формуються повідомлення MT191 для відправки за допомогою програмного комплексу Back_office. Вид файлу, що містить сформований повідомлення MT191, наведено на рис. 2.12.

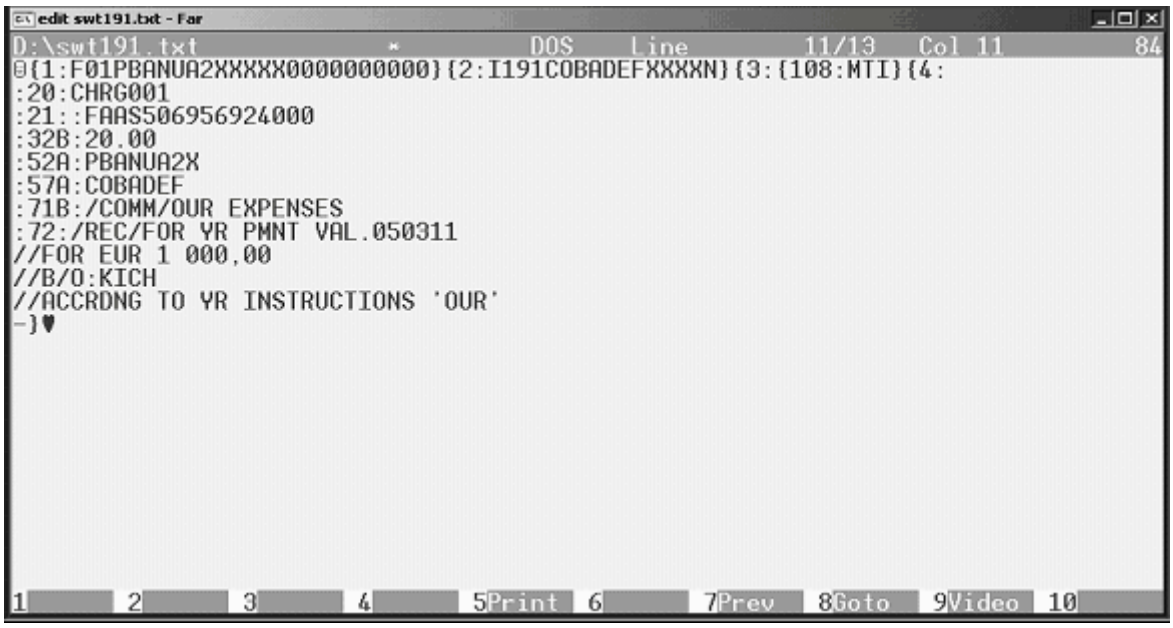

Рис. 2.11. Сформоване повідомлення MT191

Також, в файловому вигляді формується інформація, отримана з розслідувань. В отриманому файлі, надаються довідкові дані про шукані SWIFT - повідомлення. Ескіз файлу наведено на рис. 2.12.

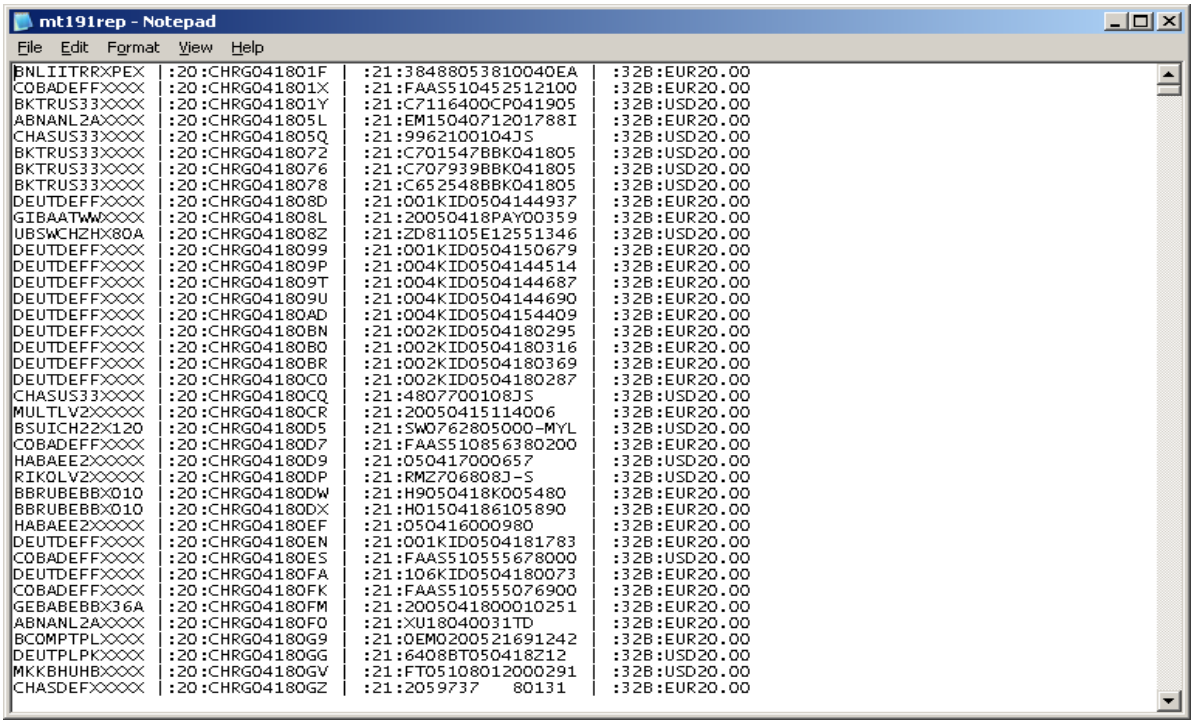

Рис. 2.12. Файл, що містить результати пошуку

У діалоговому режимі виводиться інформація, що стосується розслідувань, що проводяться співробітниками відділу по нез'ясованим комісіям.

Результат роботи представлений у вигляді діалогового вікна, в якому користувачеві представлена ключова інформація, яка однозначно ідентифікує SWIFT - повідомлення. Ескіз форми наведено на рис. 2.13.

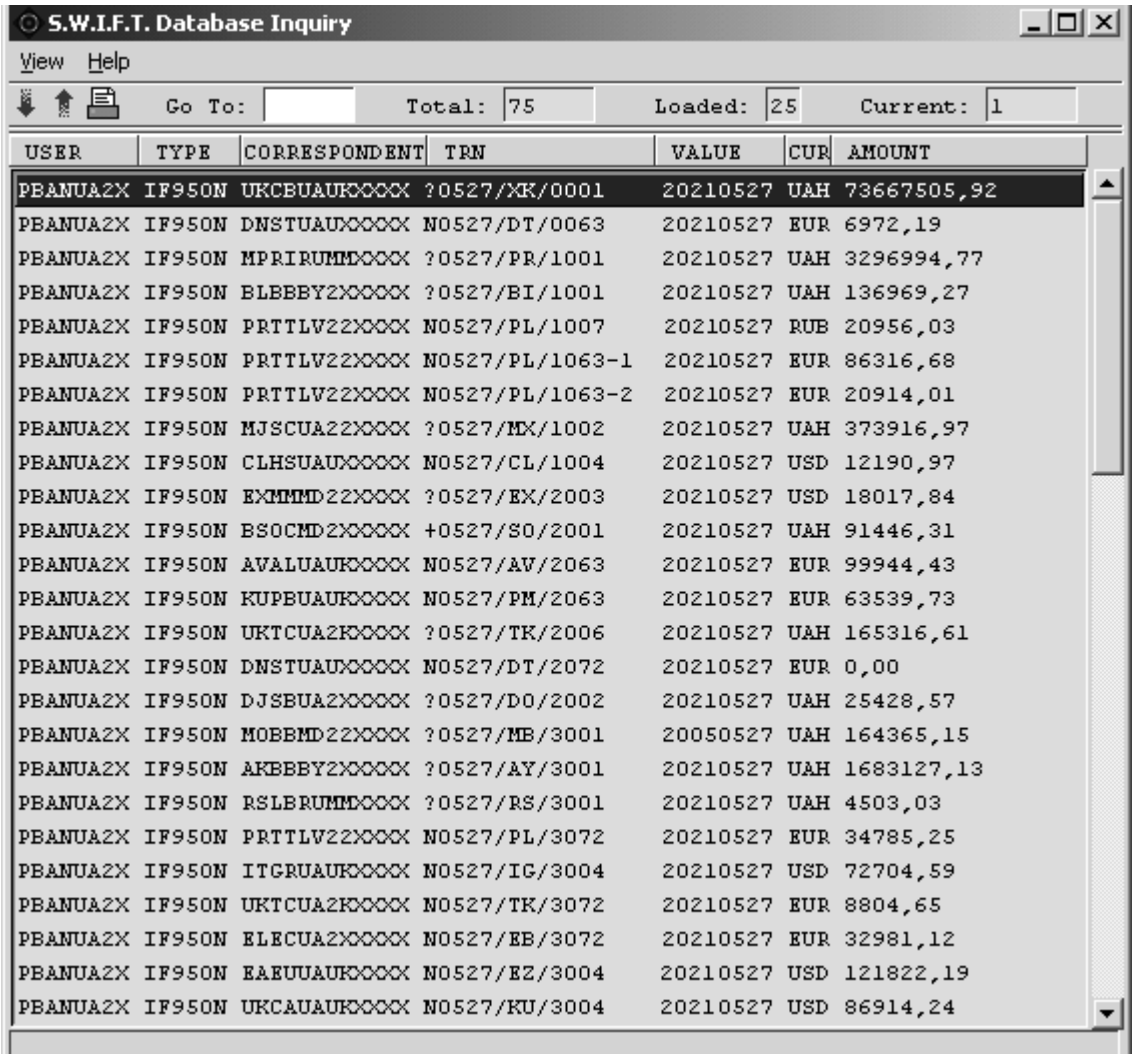

Рис. 2.13. Діалогове вікно результатів пошуку

РОЗДІЛ 3
ЕКОНОМІЧНИЙ РОЗДІЛ

3.1. Розрахунок трудомісткості та вартості розробки програмного продукту

Вхідні дані:

- передбачуване число операторів – 1000;
- коефіцієнт складності програми – 2;
- коефіцієнт корекції програми в ході її розробки – 0,08;
- годинна заробітна плата програміста, грн / год – 40.

Нормування праці в процесі створення ПЗ істотно ускладнено в силу творчого характеру праці програміста. Тому трудомісткість розробки ПЗ може бути розрахована на основі системи моделей з різною точністю оцінки.

Трудомісткість розробки ПЗ можна розрахувати за формулою:

$$t = t_u + t_a + t_n + t_{отл} + t_д, \text{ людино-годин,} \quad (3.1)$$

де to - витрати праці на підготовку й опис поставленої задачі (приймається 50),

ти - витрати праці на дослідження алгоритму рішення задачі,

та - витрати праці на розробку блок-схеми алгоритму,

тп - витрати праці на програмування по готовій блок-схемі,

тотл - витрати праці на налагодження програми на ЕОМ,

тд - витрати праці на підготовку документації.

Складові витрати праці визначаються через умовне число операторів у ПЗ, яке розробляється.:

- умовне число операторів (підпрограм):

$$Q = q \times C \times (1+ p) \text{ , людино-годин,} \qquad (3.2)$$

де q - передбачуване число операторів,

С - коефіцієнт складності програми,

р - коефіцієнт кореляції програми в ході її розробки.

$$Q = 1000 \cdot 2 \cdot (1 + 0{,}08) = 2808 \text{ людино-годин.}$$

Витрати праці на вивчення опису задачі ti визначається з урахуванням уточнення опису і кваліфікації програміста:

$$t_u = \frac{QB}{(75...85)K} \text{ , людино-годин,} \qquad (3.3)$$

де В - коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі; В=1.2 ... 1.5,

k - коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності.

Витрати праці на розробку алгоритму рішення задачі:

$$t_u = \frac{2808 \cdot 1{,}2}{80 \cdot 0{,}8} = 52 \text{ , людино-годин.}$$

Витрати на складання програми по готовій блок-схемі:

$$t\alpha = \frac{Q}{(20...25)K} \quad \text{людино-годин.} \qquad (3.4)$$

$$t_a = \frac{2808}{20 \cdot 0{,}8} = 175 \text{ людино-годин.}$$

Витрати на складання програми по готовій блок-схемі:

$$t_n = \frac{Q}{(20...25)K} \quad \text{людино-годин.} \tag{3.5}$$

$$t_n = \frac{2808}{25 \cdot 0{,}8} = 140 \text{ людино-годин.}$$

Витрати праці на налагодження програми на ЕОМ:

$$t_{отл} = \frac{Q}{(4...5)K} \quad \text{людино-годин.} \tag{3.6}$$

$$t_{отл} = \frac{2808}{4 \cdot 0{,}8} = 877 \text{ людино-годин.}$$

- за умови комплексного налагодження завдання:

$$t^{к}_{отл} = 1{,}2 \cdot t_{отл}; \tag{3.7}$$

$$t_{отл} = 877 * 1{,}2 = 1053$$

Витрати праці на підготовку документації:

$$t_д = t_{др} + t_{до}, \text{ людино-годин,} \tag{3.8}$$

де tдр - трудомісткість підготовки матеріалів і рукопису.

$$t_{др} = \frac{Q}{(15...20)K}, \text{ людино-годин.} \tag{3.9}$$

$$t_{др} = \frac{2808}{15 \cdot 0{,}8} = 234 \text{ людино-годин,}$$

tдо – трудомісткість редагування, печатки й оформлення документації

$$t_{до} = 0{,}75 \cdot t_{др}, \text{ людино-годин.} \quad (3.10)$$

$$tdo = 0{,}75 \cdot 234 = 175$$

$$t_{д} = 234 + 175 = 409 \text{ людино-годин.}$$

Отримуємо трудомісткість розробки програмного забезпечення:

$$t = 50 + 52 + 175 + 140 + 1053 + 409 = 1861 \text{ людино-годин.}$$

3.2. Розрахунок витрат на створення програми

Витрати на створення ПЗ Кпо включають витрати на заробітну плату виконавця програми Зз/п і витрат машинного часу, необхідного на налагодження програми на ЕОМ:

$$К_{ПО} = З_{ЗП} + З_{МВ}, \text{ грн,} \quad (3.11)$$

Заробітна плата виконавців визначається за формулою:

$$З_{ЗП} = t \cdot С_{ПР}, \text{ грн,} \quad (3.12)$$

де t - загальна трудомісткість, людино-годин,

Спр - середня годинна заробітна плата програміста, грн/година

$$З_{ЗП} = 1861 \cdot 30 = 43755 \text{ грн.}$$

Вартість машинного часу, необхідного для налагодження програми на ЕОМ:

$$З_{МВ} = t_{отл} \times С_{М}, \text{ грн,} \quad (3.13)$$

де tотл - трудомісткість налагодження програми на ЕОМ, год.

Смч - вартість машино-години ЕОМ, 5 грн/год.

Визначені в такий спосіб витрати на створення програмного забезпечення є частиною одноразових капітальних витрат на створення АСУП.

$$З_{МВ} = 810 \times 5 = 4050 \text{ грн.}$$

$$Кпо = 43755 + 4050 = 47805 \text{ грн.}$$

Очікуваний період створення ПЗ:

$$T = \frac{t}{B_k \cdot F_p} \text{ міс.} \qquad (3.13)$$

де Bk - число виконавців,

Fp - місячний фонд робочого часу (при 40 годинному робочому тижні Fp=176 годин).

$$T = \frac{2160}{1 \cdot 176} = 8{,}3 \text{ міс.}$$

Висновок: були проведені обчислення трудомісткості та витрат для розробки програмного забезпечення. Для створення даної інформаційної системи знадобиться 8,3 місяця, трудомісткість розробки ПЗ – 1861 людино-годин, а витрати на її створення програмного забезпечення – 47805 грн.

ВИСНОВКИ

У даній кваліфікаційній роботі була розроблена інформаційна система для обробки та обліку комісій комерційних банків.

Розроблена система працює спільно з програмним комплексом Back_office, який на вимогу розробників, функціонує під ОС Windows, а база даних працює на Sybase SQL-AnyWhere Egine. Для реалізації клієнтської частини інформаційної системи використана середа візуального програмування Microsoft Visual 2010.

Розробка даного продукту була необхідна для швидкої і ефективної обробки повідомлень про кредитування MT103, автоматизації формування повідомлень про сплату комісій третіх банків MT191, а також для пошуку банків, які відмовляються сплачувати комісії, з метою розслідування і усунення причин таких відмов з повторним повідомленням банків про необхідність оплати комісії.

Дана робота підвищує надійність і швидкість обробки фінансових повідомлень, а також знижує навантаження на персонал відділу шляхом автоматизації операторської роботи.

Впровадження даного програмного продукту дозволить зменшити час, що витрачається на з'ясування неточностей в розслідуваних платежах та на занесення даних в корпоративну базу даних. Таким чином, збільшується вірогідність і своєчасність наданої інформації, а, отже збільшуються доходи банку.

В економічному розділі були проведені обчислення трудомісткості та витрат для розробки програмного забезпечення. Для створення даної інформаційної системи знадобиться 8,3 місяця, трудомісткість розробки ПЗ – 1861 людино-годин, а витрати на її створення програмного забезпечення - 47805 грн.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ананьєв, О.М. Інформаційні системи і технології в комерційній діяльності [Текст]: підручник / О.М. Ананьєв, В.М. Білик, Я.А. Гончарук. – Львів: Новий Світ-2000, 2006. – 584 с.

2. Антонов, В.М. Фінансовий менеджмент: сучасні інформаційні технології [Текст]: навчальний посібник / В.М. Антонов, Г.К. Яловий; ред. В.М. Антонов; Мін-во освіти і науки України, КНУ ім. Т.Г. Шевченка. – К.: ЦНЛ, 2005. – 432 с.

3. Арбузов С. Г., Колобов Ю. В., Міщенко В. І., Науменкова С. В. СВІФТ // Банківська енциклопедія. - Київ : Центр наукових досліджень Національного банку України : Знання, 2011. - 504 с. - (Інституційні засади розвитку банківської системи України). - ISBN 978-966-346-923-2.

4. Архангельский А.Я. C++ Builder 6. Справочное пособие. Книга 1. Язык C++. – М.: Бином-Пресс, 2002 г. – 544 с.: ил. ISBN 978-5-9518-0229-3.

5. Бібліографічний запис. Бібліографічний опис. Загальні вимоги та правила складання : (ГОСТ 7.1-2003, IDT) : ДСТУ ГОСТ 7.1:2006. – Чинний з 2007–07–01. – К. : Держспоживстандарт України, 2007. – 47 с. – (Система стандартів з інформації, бібліотечної та видавничої справи) (Національний стандарт України).

6. Бусигін Б.С., Коротенко Г.М., Коротенко Л.М. Прикладна інформатика. Підручник для студентів комп'ютерних спеціальностей. – Дніпропетровськ: Видавництво НГУ, 2004. – 559 с. URL: http://www.programmer.dp.ua/book-ua-k01.php. дата звернення: 15.03.2019.

7. Бусыгин Б.С., Дивизинюк М.М., Коротенко Г.М., Коротенко Л.М. Введение в современную информатику. Учебник. – Севастополь: Издательство СНУЯЭиП, 2005. – 644 с. / URL: http://www.programmer.dp.ua/book-ru-k02.php. дата звернення: 15.01.2018.

8. Документація. Звіти у сфері науки і техніки. Структура і правила оформлення : ДСТУ 3008-95. – Чинний від 1996–01–01. – К. : Держстандарт України, 1996. – 39 с.

9. ДСТУ 2394-94 Інформація та документація. Комплектування фонду, бібліографічний опис, аналіз документів. Терміни та визначення. – Чинний від 01.01.1995. - Київ : Держстандарт України, 1994. – 88 с.

10. Методичні вказівки з виконання економічного розділу в дипломних проектах студентів спеціальності “Комп’ютерні системи ” / Укладачі О.Г. Вагонова, Нікітіна О.Б. Н.Н. Романюк – Дніпропетровськ: Національний гірничий університет. – 2013. – 23с.

11. Методичні рекомендації до виконання кваліфікаційних робіт бакалаврів напряму підготовки 122 «Комп’ютерні науки» галузі знань 12 Інформаційні технології/, Л.М. Коротенко , О.С. Шевцова; Нац. гірн. ун-т. – Д : ДВНЗ НГУ, 2019. – 65 с.

12. Олійник, А.В. Інформаційні системи і технології у фінансових установах [Текст]: навчальний посібник / А.В. Олійник, В.М. Шацька. – Львів: Новий Світ-2000, 2006. – 436 с.

13. Офіційний сайт середовища розробки Visual Studio Code URL: https://code.visualstudio.com/docs. дата звернення: 12.05.2021.

14. Пасічник, В.В. Організація баз даних та знань/ В.В.Пасічник, В.А.Резніченко.–К.: Видавнича група BHV, 2006.– 384 сю:іл.– ISBN966-552-156-X

15. Румянцев, М.И. Информационные системы и технологии финансово-кредитных учреждений [Текст]: учебное пособие для вузов / М.И. Румянцев; Западнодонбасский ин-т экономики и управления. – Днепропетровск: ИМА-пресс, 2006. – 482 с.

16. Системы и технологии: приложения в экономике и управлении: Кн. 6 [Текст]: учебное пособие / Мин-во образования и науки Украины, Донецкий нац. ун-т; ред. Ю.Г. Лысенко. – Донецк: Юго-Восток, 2004. – 377 с.

17. Скорочення слів в українській мові у бібліографічному описі. Загальні правила та вимоги : ДСТУ 3582-97. – Чинний від 1998–07–01. – К. : Держстандарт України, 1998. – 24 с. – (Державний стандарт України).

18. Степанов Ю.Л. Разработка приложений баз данных для СУБД SYBASE SQL ANYWHERE.

19. Тейксера, Стив, Пачеко, Ксавье. Руководство разработчика, том 1. Основные методы и технологии программирования.: Пер. с. англ.: Уч. пос. – М.: Издательский дом «Вильямс», 2000. – 832 с.: ил. – Парал. тит. англ.

20. Тейксера, Стив, Пачеко, Ксавье. Руководство разработчика, том 2. Разработка компонентов и программирование баз данных.: Пер. с. англ.: Уч. пос. – М.: Издательский дом «Вильямс», 2000. – 992 с.: ил. – Парал. тит. англ.

ДОДАТОК А

КОД ПРОГРАМИ

```
/* SQL Anywhere 3/1/17*/set option date_format='Mmm dd yyyy hh:mmaa'goset option
date_order = 'MDY'goset option scale = 2godrop table bicgocreate table bic(bic_id integer not null
primary key
DEFAULT autoincrement,
bic char(11) not null
unique,
stat char(1)
default 'C'
check (stat in ('C','D','5'))
)
go
grant select on bic to publicgodrop table curcgocreate table curc
(
curc_id integer not null primary key
DEFAULT autoincrement,
curc char(3) not null,
summa char(17) default '20,00' not null,
Comments varchar(50) null,
bic_id integer null references bic(bic_id))gogrant select on curc to publicgo
drop table mt_103_rcvdgocreate table mt_103_rcvd(
id_103 integer not null primary key
DEFAULT autoincrement,
f_20 char(16) not null,
vald char(8) not null,
curc char(3) not null,
amnt char(17) not null,sndr char(12) not null,
text varchar(32767) not null,
ms_dbky char(18) not null
)go
grant select on mt_103_rcvd to publicgodrop table mt_910gocreate table mt_910(
id_910 integer not null primary key
DEFAULT autoincrement,f_20 char(16) not null,
f_21 char(16) not null
default 'NON REFF',
vald char(8) not null,
curc char(3) not null,
amnt char(17) not null,sndr char(12) not null,
f_72 varchar(210) null,
ms_dbky char(18) not null)gogrant select on mt_910 to publicgo
drop table mt_202go
create table mt_202(
id_202 integer not null primary key
DEFAULT autoincrement,f_20 char(16) not null,
f_21 char(16) not null
default 'NON REFF',
vald char(8) not null,
```

```
curc char(3) not null,
amnt char(17) not null,sndr char(12) not null,
f_72 varchar(210) null,
ms_dbky char(18) not null)gogrant select on mt_202 to publicgo
drop table mt_950go
create table mt_950(
id_950 integer not null primary key
DEFAULT autoincrement,f_20 char(16) not null,
f_21 char(16) not null
default 'NON REFF',
f_61 varchar(98) null,
ms_dbky char(18) not null)gogrant select on mt_950 to publicgo
drop table mt_n9ngo
create table mt_n9n(
n9n_id integer not null primary key
DEFAULT autoincrement,f_20 char(16) not null,
f_21 char(16) not null
default 'NON REFF',
text varchar(32767) not null,
ms_dbky char(18) not null)gogrant select on mt_n9n to publicgo
drop table mt_191_sendgocreate table mt_191_send(
id_191 integer not null primary key
DEFAULT autoincrement,
f_20 char(16) not null,
f_21 char(16) not null,
vald char(8) not null,
curc char(3) not null,
amnt char(17) not null,sndr char(12) not null,
stat char(1) default 'N' not null
check (stat in ('N','P')),
ms_dbky char(18) not null,
f_72 varchar(210) null,
id_103 integer references mt_103_rcvd(id_103) null,
id_910 integer references mt_910(id_910) null,
id_202 integer references mt_202(id_202) null,
id_950 integer references mt_950(id_950) null,
n9n_id integer references mt_n9n(n9n_id) null ) gogrant select on mt_191_send to public
go
drop table dep go
create table dep
(
id_dep integer not null primary key
default autoincrement,
DepName varchar(30) null unique,
Comments varchar(225) null,
)
go
drop table empl
go
create table empl
(
        id_empl integer not null primary key
```

```
        default autoincrement,
        FName varchar(20) null,
        LName varchar(20) null,
        id_dep integer references dep(id_dep)
)
go
drop table ugroup
go
create table ugroup
(
        id_group integer not null primary key
        default autoincrement,
        GName varchar(30) not null unique
        default ('user')
        check (GName in('user','admin','guest')),
        Comments varchar(255) null
)
go
drop table rights
go
create table rights
(
 id_right integer not null primary key
 default autoincrement,
 flags char(5) not null
 default 'R'
 check (flags in ('R','RW','IUD','RWUID')),
 id_group integer references ugroup(id_group),
 id_secr integer references secrets(id_secr)
)
go
create table secrets
(
        id_secr integer not null primary key
        default autoincrement,
        uid varchar(10) not null unique
        default ('chrg_use'),
        pwd varchar(10)not null
        default ('chrg_use'),
        datein char(8) null,
        id_empl integer references empl(id_empl)
)
go
create unique index bicid on bic(bic)gocreate index camnt_103 on mt_103_rcvd (curc,amnt)go
create index f_20_103 on mt_103_rcvd (f_20)go
create index f_20_191 on mt_191_send (f_20)gocreate index f_20_910 on mt_910 (f_20)go
create index f_20_202 on mt_202 (f_20)go
create index f_20_950 on mt_950 (f_20)go
create index f_20_n9n on mt_n9n (f_20)go
```

Скрипт хранимых процедур для базы chrgas

```
/* SQL Anywhere 3/1/96*/set option date_format='Mmm dd yyyy hh:mmaa'goset option
date_order = 'MDY'goset option scale = 2go
drop procedure add_mt_103_rcvd
go
create procedure add_mt_103_rcvd(
in inf_20 char(16),
in invald char(8),
in incurc char(3),
in inamnt char(17),
in insndr char(12),
in intext varchar(32767),
in inms_dbky char(18))
begin
insert into mt_103_rcvd(f_20,vald,curc,amnt,sndr,text,ms_dbky)
values(inf_20,invald,incurc,inamnt,insndr,intext,inms_dbky)
end
go
drop procedure add_mt_910
go
create procedure add_mt_910
(
in inf_20 char(16),
in inf_21 char(16),
in invald char(8),
in incurc char(3),
in inamnt char(17),
in insndr char(12),
in inf_72 varchar(210),
in inms_dbky char(18)
)
begin
insert into mt_910(f_20,f_21,vald,curc,amnt,sndr,f_72,ms_dbky)
values(inf_20,inf_21,invald,incurc,inamnt,insndr,inf_72,inms_dbky)
end
go
drop procedure add_mt_202
go
create procedure add_mt_202(
in inf_20 char(16),
in inf_21 char(16),
in invald char(8),
in incurc char(3),
in inamnt char(17),
in insndr char(12),
in inf_72 varchar(210),
in inms_dbky char(18))
begin
insert into mt_202(f_20,f_21,vald,curc,amnt,sndr,f_72,ms_dbky)
values(inf_20,inf_21,invald,incurc,inamnt,insndr,inf_72,inms_dbky)
end
go
```

```
drop procedure add_mt_950
go
create procedure add_mt_950(
in inf_20 char(16),
in inf_21 char(16),
in inf_61 varchar(98),
in inms_dbky char(18))
begin
insert into mt_950(f_20,f_21,f_61,ms_dbky)
values(inf_20,inf_21,inf_61,inms_dbky)
end
go
drop procedure add_mt_n9n
go
create procedure add_mt_n9n(
in inf_20 char(16),
in inf_21 char(16),
in intext varchar(32767),
in inms_dbky char(18)
)
begin
insert into mt_n9n(f_20,f_21,text,ms_dbky)
values(inf_20,inf_21,intext,inms_dbky)
end
go
drop procedure add_mt_191_send
go
create procedure add_mt_191_send(
in inf_20 char(16),
in inf_21 char(16),
in invald char(8),
in incurc char(3),
in inamnt char(17),
in insndr char(12),
in instat char(1),
in inms_dbky char(18),
in inf_72 varchar(210)
)
begin
insert into mt_191_send(f_20,f_21,vald,curc,amnt,sndr,stat,ms_dbky,f_72)
values(inf_20,inf_21,invald,incurc,inamnt,insndr,instat,inms_dbky,inf_72)
end
go
drop procedure add_dep
go
create procedure add_dep(
in inDepName varchar(30),
in inComments varchar(225)
)
begin
        insert into dep(DepName,Comments)
        values(inDepName,inComments)
```

```
end
go
drop procedure upd_dep
go
create procedure upd_dep(
in oldDepName varchar(30),
in inDepName varchar(30),
in inComments varchar(225)
)
begin
        declare in_id integer;
        select id_dep into in_id
        from dep
        where  DepName = oldDepName;
        update dep
        set DepName=inDepName,Comments=inComments
        where id_dep = in_id
end
go
drop procedure del_dep
go
create procedure del_dep(
in inDepName varchar(30)
)
begin
declare in_id integer;
select id_dep into in_id
from dep
        where DepName=inDepName;
delete from dep
where id_dep = in_id
end
go
drop procedure add_empl
go
create procedure add_empl(
in inFName varchar(20),
in inLName varchar(20),
in inDepName varchar(30)
)
begin
        declare in_id integer;
        select id_dep into in_id
        from dep
        where DepName=inDepName;
        insert into empl(FName,LName,id_dep)
        values(inFName,inLName,in_id)
end
go
drop procedure upd_empl
go
create procedure upd_empl(
```

```
in inid_empl integer,
in inFName varchar(20),
in inLName varchar(20),
in inDepName varchar(30))
begin
        declare inid integer;
        select id_dep into inid
        from dep
        where DepName=inDepName;
        update empl
        set FName=inFName,LName=inLName,id_dep=inid
        where id_empl = inid_empl;
end
go
drop procedure del_empl
go
create procedure del_empl(
in inid_empl integer
)
begin
        delete from empl
        where id_empl = inid_empl
end
go
drop procedure add_ugroup
go
create procedure add_ugroup(
in inGName varchar(30),
in inComments varchar(255)
)
begin
        insert into ugroup(GName,Comments)
        values(inGName,inComments)
end
go
drop procedure upd_ugroup
go
create procedure upd_ugroup(
in oldGName varchar(30),
in inid_group integer,
in inGName varchar(30),
in inComments varchar(255)
)
begin
        declare in_id integer;
        select id_group into in_id from ugroup
        where GName=oldGName;
        update ugroup
        set GName = inGName,Comments = inComments
        where id_group=in_id
end
go
```

```
drop procedure del_ugroup
go
create procedure del_ugroup(
in inid_group integer
)
begin
        delete from ugroup
        where id_group = inid_group
        end
go
drop procedure add_rights
go
create procedure add_rights(
in inflags char(5),
in inGName varchar(30),
in inUid varchar(10)
)
begin
        declare in_id integer;
        declare inid_secr integer;
        select id_group into in_id from ugroup
        where GName=inGName;
        select id_secr into inid_secr from secrets
        where uid=inUid;
        insert into rights(flags,id_group,id_secr)
        values(inflags,in_id,inid_secr)
end
go
drop procedure upd_rights
go
create procedure upd_rights(
in inflags char(5),
in inGName varchar(30)
)
begin
        declare in_id integer;
        declare in_id_r integer;
        select id_group into in_id
        from ugroup
        where GName = inGName;
        select id_right into in_id_r
        from rights
        where id_group = in_id;
        update rights set flags=inflags
        where id_right = in_id_r
end
go
drop procedure del_rights
go
create procedure del_rights(
in inid_right integer
)
```

```
begin
        delete from rights
        where id_right=inid_right
end
go
drop procedure add_secrets
go
create procedure add_secrets(
in inuid varchar(10),
in inpwd varchar(10),
in indatein char(8),
in inid_empl integer
)
begin
        insert into secrets(uid,pwd,datein,id_empl)
        values(inuid,inpwd,indatein,inid_empl)
end
go
drop procedure upd_secrets
go
create procedure upd_secrets(
in inGName varchar(30),
in inid_secr integer
)
begin
        declare inid_right integer;
        declare in_id integer;
        select id_group into in_id
        from ugroup
        where GName = inGName;
        select id_right into inid_right
        from rights
        where id_group = in_id;
   update secrets set
   id_right=inid_right
   where id_secr=inid_secr
end
go
drop procedure del_secrets
go
create procedure del_secrets(
in inid_secr integer
)
begin
        delete from secrets
        where id_secr=inid_secr
end
go
drop procedure add_bic
go
create procedure add_bic
(
```

```
inbic char(11),
instat char(1)
)
begin
        insert into bic(bic,stat)
        values(inbic,instat)
end
go
drop procedure del_bic
go
create procedure del_bic
(
inbic char(11)
)
begin
        delete from bic
        where bic = inbic;
end
go
drop procedure upd_bic
go
create procedure upd_bic
(
inbic_id integer,
inbic char(11),
instat char(1)
)
begin
        update bic set bic = inbic,stat=instat
        where bic_id=inbic_id;
end
go
drop procedure add_curc
go
create procedure add_curc
(
incurc char(3),
insumma char(17),
inComments varchar(50),
inbic char(11)
)
begin
        declare in_id integer;
        select bic_id into in_id
        from bic
        where bic = inbic;
        insert into curc(curc,summa,Comments,bic_id)
        values(incurc,insumma,inComments,in_id)
end
go
drop procedure upd_curc
go
```

```
create procedure upd_curc
(
incurc_id integer,
incurc char(3),
insumma char(17),
inComments varchar(50),
inbic char(11)
)
begin
        declare in_id integer;
        select bic_id into in_id from bic
        where bic = inbic;
        update curc
        set curc=incurc,summa=insumma,Comments=inComments,bic_id=in_id
        where curc_id=incurc_id
end
go
drop procedure del_curc
go
create procedure del_curc
(
incurc_id integer
)
begin
        delete from curc
        where curc_id = incurc_id;
end
go
create trigger before_del_bic
before delete on bic
referencing old as del_bic_id
for each row
begin
        delete from curc
        where curc.bic_id=del_bic_id;
end
go
create trigger after_update_bic
after update of bic_id on bic
referencing old as old_bic_id
new as new_bic_id
for each row
begin
        update curc
        set curc.bic_id=new_bic_id
        where curc.bic_id=old_bic_id
end
go
create trigger before_del_secrets
before delete on secrets
referencing old as del_empl_id
for each row
```

```
begin
        delete from empl
        where empl.id_empl=del_empl_id
end
go
create trigger before_del_empl
before delete on empl
referencing old as del_dep_id
for each row
begin
        delete from dep
        where dep.id_dep=del_dep_id
end
go
```

Скрипт хранимых процедур базы Back_off

```
create procedure read_103(in skey varchar(18))
result ("ms_trnf" char(16),"ms_vald" char(8),"ms_curc" char(3),"ms_amnt" char(17),"ms_sndr"
char(12),"ms_dbky" char(18), "ms_text" char(500))
begin
        select distinct ms_trnf,ms_vald,ms_curc,ms_amnt,ms_sndr,ms_dbky,ms_text from ms_
        where ms_stat = 'R'
        and ms_dirc = 'O'
        and ms_mtyp = '103'
        and ms_dbky like skey;
end

create procedure read_103_all(in skey varchar(18))
result ("ms_text" char(574))
begin
        select distinct ms_trnf+"->"+ms_vald+"->"+ms_curc+"->"+ms_amnt+"->"+ms_sndr+"-
>"+ms_dbky+"->"+ms_text from ms_
        where ms_stat = 'R'
        and ms_dirc = 'O'
        and ms_mtyp = '103'
        and ms_dbky like skey;
end

/*по db_key возвращает всё сообщение, без его головы (первые 500 байт)*/
create procedure rmte(in skey varchar(18),out rslt_text varchar(32767))
begin
   declare err_notf
   EXCEPTION FOR SQLSTATE '02000';
   declare tmp_text varchar(32767);
   declare cselmx cursor for
        select mx_text from mx_
        where mx_dbky=skey
        order by mx_sgmt asc;
   set rslt_text = '';
   open cselmx;
   mxloop:
   loop
```

```
        fetch next cselmx
            into tmp_text;
        if sqlstate = err_notf then
            leave mxloop;
        end if;
        set rslt_text=rslt_text+tmp_text;
    end loop mxloop;
close cselmx;
end
create procedure rmte_sel(in skey varchar(20))
begin
    declare rslt_text varchar(32767);
    declare err_notf
    EXCEPTION FOR SQLSTATE '02000';
    declare tmp_text char(500);
    declare cselmx cursor for
          select mx_text from mx_
          where mx_dbky=skey
          order by mx_sgmt asc;
    set rslt_text = '';
    open cselmx;
    mxloop:
    loop
        fetch next cselmx
            into tmp_text;
        if sqlstate = err_notf then
            begin
                set rslt_text=rslt_text + '';
            leave mxloop;
            end
        end if;
        set rslt_text=rslt_text+tmp_text;
    end loop mxloop;
close cselmx;
select rslt_text;
end
create procedure read_103_all(in skey varchar(20))
result (fnd_dbky varchar(20), rslt_text char(600),end_text varchar(32767))
begin
          select distinct ms_dbky,ms_text,(select * from rmte_sel(ms_dbky))as end_text from ms_
          where ms_stat = 'R'
          and ms_dirc = 'O'
          and ms_mtyp = '103'
          and ms_dbky like skey;
end
create procedure read_103_all(in skey varchar(20))
result (fnd_dbky varchar(20), all_text varchar(32767))
begin
          select distinct ms_dbky,ms_text+(select * from rmte_sel(ms_dbky)) from ms_
          where ms_stat = 'R'
          and ms_dirc = 'O'
          and ms_mtyp = '103'
```

```
        and ms_dbky like skey;
end

create procedure read_103_all(in skey varchar(20))
result (fnd_dbky varchar(20), all_text varchar(32767))
begin
   declare rslt_text varchar(32767);
        select distinct ms_dbky,ms_text+(select rslt_text from rmte_sel(ms_dbky)) from ms_
        where ms_stat = 'R'
        and ms_dirc = 'O'
        and ms_mtyp = '103'
        and ms_dbky like skey;
end
SELECT SYSPROCEDURE.proc_defn
FROM SYS.SYSPROCEDURE
where SYSPROCEDURE.proc_name = 'mt_103_rc';
output to d:\proc.sql format ascii
create procedure mt103rc(in skey varchar(18),out text varchar(32727))
begin
        select distinct ms_text into text from ms_
        where ms_stat = 'R'
        and ms_dirc = 'O'
        and ms_mtyp = '103'
        and ms_dbky like skey;
end
```

Модуль для взаимодействия с базой данных программы Back_office.

```
using System;
using System.Data;
using iAnywhere.Data.AsaClient;
using System.Windows.Forms;
using System.Diagnostics;
using System.Collections.Specialized;
namespace ComGenerator
{
        /// <summary>
        /// Summary description for CGetFromTs.
        /// </summary>
        public class CGetFromTs
        {
                private string str_conn;
                private string str_cmd;
                private StringCollection str_rslt;
                //Anywhere data types
                private AsaCommand asa_cmd;
                private AsaConnection asa_conn;
                private AsaDataAdapter asa_adapter;
                //Common data types
                private DataSet ds;
                private DataTable dt;
                public CGetFromTs()
                {
```

```
        str_conn = "Data Source=TS;UID=TS_DBA;PWD=TS_DBA";
str_cmd = "select * from sys.systable where creator=1";
                        }
public CGetFromTs(string cmd)
{
                str_conn = "Data Source=TS;UID=TS_DBA;PWD=TS_DBA";
        str_cmd = System.String.Copy(cmd);
                }
public bool SetConnection()
{
        bool flag=false;
        try
        {
                asa_conn = new AsaConnection(str_conn);
                asa_conn.Open();
                flag = true;
        }
        catch(AsaException ex )
        {
                MessageBox.Show( ex.Errors[0].Source + " : "
                        + ex.Errors[0].Message + " (" +
                        ex.Errors[0].NativeError.ToString() + ")",
                        "Failed to connect" );
        }
        return flag;
}
public StringCollection Get_103(string str_dbkey)
{
        asa_conn = new AsaConnection(str_conn);
        try
        {
                asa_conn.Open();
                asa_cmd = new AsaCommand("read_103",asa_conn);
                asa_cmd.CommandType = CommandType.StoredProcedure;
                asa_cmd.Parameters.Add("@dbky",AsaDbType.Char,18);

                asa_cmd.Parameters[0].Value = str_dbkey;
                asa_adapter = new AsaDataAdapter(asa_cmd);
                AsaDataReader rdr = asa_cmd.ExecuteReader();
                ds = new DataSet();
                ds.EnforceConstraints = false;
                int i = asa_adapter.Fill(ds);
                ds.EnforceConstraints = true;
                str_rslt = new StringCollection();
                while(rdr.Read())
                {
                        str_rslt.Add(rdr[0] + "#" + rdr[1] + "#"+ rdr[2] + "#"
                                + rdr[3] + "#"+ rdr[4] + "#"+ rdr[5] + "#"+ rdr[6]);
                }
        }
        catch(AsaException asaex)
        {
```

```
                LogException(asaex);
        }
        catch(Exception ex)
        {
                throw ex;
        }
        finally
        {
                asa_conn.Close();
        }
        return str_rslt;
}
public string Get_103_all(string str_dbkey)
{
        string str_rslt = new string(' ',1500);
        str_rslt = str_rslt.Trim();
        asa_conn = new AsaConnection(str_conn);
        try
        {
                asa_conn.Open();
                asa_cmd = new AsaCommand("rmte",asa_conn);
                asa_cmd.CommandType = CommandType.StoredProcedure;
                asa_cmd.Parameters.Add("@dbky",AsaDbType.Char,18);
                asa_cmd.Parameters[0].Value = str_dbkey;
                if((str_rslt = (string)asa_cmd.ExecuteScalar()) == null )
                        str_rslt = " ";
        }
        catch(AsaException asaex)
        {
                LogException(asaex);
        }
        catch(Exception ex)
        {
                throw ex;
        }
        finally
        {
                asa_conn.Close();
        }
        str_rslt = str_rslt.Trim();
        return str_rslt;
}
public bool CloseCon()
{
        asa_conn.Close();
        return true;
}
private void LogException(AsaException exc)
{
        EventLog el = new EventLog();
        el.Source = "ComGenerator";
        string strMessage;
```

```
                strMessage = "Exception Number : " + exc.Errors.Count +exc.Message + "
has ocurred";
                el.WriteEntry(strMessage);
                foreach(AsaError asaerr in exc.Errors)
                {
                        strMessage = "Message" + asaerr.Message+
                                "Source " + asaerr.Source;
                        el.WriteEntry(strMessage);
                }
        }
        public string GetRow(DataRow row)
        {
                DataTable tbl = row.Table;
                string str = new string(' ',(tbl.Columns.Count)*255);
                str = str.Trim();
                foreach(DataColumn col in tbl.Columns)
                {
                        str = str+row[col]+"#";
                }
                return str;
        }
        public DataSet CreateDataSet()
        {
                DataSet ds = new DataSet();
                DataTable tbl;
                DataColumn col;
                ForeignKeyConstraint fk;
                //Add Table mt_103_rcvd in User ds
                tbl = ds.Tables.Add("mt_103_rcvd");
                col = tbl.Columns.Add("id_103",typeof(int));
                col.AutoIncrement = true;
                col.AutoIncrementSeed = -1;
                col.AutoIncrementStep = -1;
                col = tbl.Columns.Add("f_20",typeof(string));
                col.MaxLength = 16;
                col = tbl.Columns.Add("vald",typeof(string));
                col.MaxLength = 8;
                col = tbl.Columns.Add("curc",typeof(string));
                col.MaxLength = 3;
                col = tbl.Columns.Add("amnt",typeof(string));
                col.MaxLength = 17;
                col = tbl.Columns.Add("sndr",typeof(string));
                col.MaxLength = 12;
                col = tbl.Columns.Add("text",typeof(string));
                col.MaxLength = 32767;
                col = tbl.Columns.Add("ms_dbky",typeof(string));
                col.MaxLength = 18;
                col.AllowDBNull = false;
                tbl.PrimaryKey = new DataColumn[] {tbl.Columns[0]}.
```

```
                        //Add Table mt_191_send in User ds
                        tbl = ds.Tables.Add("mt_191_send");
                        col = tbl.Columns.Add("id_191",typeof(int));
                        col.AutoIncrement = true;
                        col.AutoIncrementSeed = -1;
                        col.AutoIncrementStep = -1;
                        col = tbl.Columns.Add("f_20",typeof(string));
                        col.MaxLength = 16;
                        col = tbl.Columns.Add("f_21",typeof(string));
                        col.MaxLength = 16;
                        col = tbl.Columns.Add("vald",typeof(string));
                        col.MaxLength = 8;
                        col = tbl.Columns.Add("curc",typeof(string));
                        col.MaxLength = 3;
                        col = tbl.Columns.Add("amnt",typeof(string));
                        col.MaxLength = 17;
                        col = tbl.Columns.Add("sndr",typeof(string));
                        col.MaxLength = 12;
                        col = tbl.Columns.Add("stat",typeof(string));
                        col.MaxLength = 1;
                        col.DefaultValue = "N";
                        col.AllowDBNull = false;
                        col = tbl.Columns.Add("ms_dbky",typeof(string));
                        col.MaxLength = 18;
                        col.AllowDBNull = false;
                        col = tbl.Columns.Add("id_103",typeof(int));
                        col.AllowDBNull = false;
                                        fk = new
ForeignKeyConstraint(ds.Tables[0].Columns[0],ds.Tables[1].Columns[8]);
                        ds.Tables[1].Constraints.Add(fk);
                        return ds;
                }
                public bool FillDs(int indTable,string[] strings,int col)
                {
                        bool flag = false;
                        object[] aValues;
                        foreach(string str in strings)
                        {
                                aValues = str.Split('#');
                                ds.Tables[indTable].LoadDataRow(aValues,false);
                                flag = true;
                        }
                        return flag;
                }
        }
}
```

Модули для взаимодействия с базой chargas.db
CCurc

```
using System;
using System.Data;
```

```
using System.Data.Common;
using System.Windows.Forms;
using iAnywhere.Data.AsaClient;
using System.Diagnostics;
using System.Collections.Specialized;
namespace ComGenerator
{
	/// <summary>
	/// Summary description for CCurc.
	/// </summary>
	public class CCurc
	{
		//Connection parameters
		private string str_conn;
		private string str_cmd;
		//Anywhere data types
		private AsaCommand asa_cmd;
		private AsaConnection asa_conn;
		private AsaDataAdapter asa_adapter;
		public CCurc()
		{
				str_conn = "Data Source=CHRGAS;UID=DBA;PWD=DBA";
			}
		#region Заполняет DS данными из таблицы curc
		public bool SetConnection()
		{
			bool flag=false;
			try
			{
				asa_conn = new AsaConnection(str_conn);
				asa_conn.Open();
				flag = true;
			}
			catch(AsaException ex )
			{
				MessageBox.Show( ex.Errors[0].Source + " : "
					+ ex.Errors[0].Message + " (" +
					ex.Errors[0].NativeError.ToString() + ")",
					"Failed to connect" );
			}
			return flag;
		}
		public void GetBicDetailsDataSet(DataSet ds)
		{
			try
			{
				asa_cmd = new AsaCommand();
				asa_cmd.Connection = asa_conn;
		asa_cmd.CommandText ="select curc_id,curc,summa,comments, bic_id from curc";
				asa_adapter = new AsaDataAdapter(asa_cmd);
				asa_adapter.TableMappings.Add("Table","curc_det");
				DataColumnMapping[] colMapArr;
```

```
colMapArr = new DataColumnMapping[]{new DataColumnMapping("curc_id","curc_id"),
                                    new DataColumnMapping("curc","curc"),
                        new DataColumnMapping("summa","summa"),
                        new DataColumnMapping("Comments","Comments"),
                    new DataColumnMapping("bic_id","bic_id")};
                asa_adapter.TableMappings[0].ColumnMappings.AddRange(colMapArr);
                        asa_adapter.Fill(ds);
                }
                catch(AsaException ex )
                {
                        MessageBox.Show( ex.Errors[0].Source + " : "
                                + ex.Errors[0].Message + " (" +
                                ex.Errors[0].NativeError.ToString() + ")",
                                "Failed to connect" );
                }
                catch(Exception ex)
                {
                        throw ex;
                }
                finally
                {
                        asa_conn.Close();
                }
        }
        public void GetBicDetailsDataSet(DataSet ds,int Param)
        {
                try
                {
                        asa_cmd = new AsaCommand();
                        asa_cmd.Connection = asa_conn;
                                                asa_cmd.CommandText ="select
c.curc_id,c.curc,c.summa,c.Comments,b.bic "+ "from curc c,bic b " +"where c.bic_id=b.bic_id and
b.bic_id=?";
                asa_cmd.Parameters.Add("@bic_id",AsaDbType.Integer,4,"bic_id");
                        asa_cmd.Parameters[0].Value = Param;
                        asa_adapter = new AsaDataAdapter(asa_cmd);
                        asa_adapter.TableMappings.Add("Table","curc_det");
                        DataColumnMapping[] colMapArr;
colMapArr = new DataColumnMapping[]{new DataColumnMapping("curc_id","curc_id"),
                    new DataColumnMapping("curc","curc"),
                new DataColumnMapping("summa","summa"),
                new DataColumnMapping("Comments","Comments"),
                        new DataColumnMapping("bic","bic")};
                asa_adapter.TableMappings[0].ColumnMappings.AddRange(colMapArr);
                        asa_adapter.Fill(ds);
                }
                catch(AsaException ex )
                {
                        MessageBox.Show( ex.Errors[0].Source + " : "
                                + ex.Errors[0].Message + " (" +
                                ex.Errors[0].NativeError.ToString() + ")",
                                "Failed to connect" );
```

```
                }
                catch(Exception ex)
                {
                        throw ex;
                }
                finally
                {
                        asa_conn.Close();
                }
        }
        public void GetCurcDataSet(DataSet ds)
        {
str_cmd = "select c.curc_id,c.curc,c.summa,c.comments, b.bic from curc c,bic b "+"where
c.bic_id=b.bic_id";
                try
                {
                        asa_cmd = new AsaCommand();
                        asa_cmd.Connection = asa_conn;
                        asa_cmd.CommandText =str_cmd;
                        asa_adapter = new AsaDataAdapter(asa_cmd);
                        asa_adapter.TableMappings.Add("Table","curc");
                        DataColumnMapping[] colMapArr;
                        colMapArr = new DataColumnMapping[]{new
DataColumnMapping("curc_id","curc_id"),   new DataColumnMapping("curc","curc"),    new
DataColumnMapping("summa","summa"),   new DataColumnMapping("Comments","Comments"),
  new DataColumnMapping("bic","bic")};
                asa_adapter.TableMappings[0].ColumnMappings.AddRange(colMapArr);
                        asa_adapter.Fill(ds);
                }
                catch(AsaException ex )
                {
                        MessageBox.Show( ex.Errors[0].Source + " : "
                                + ex.Errors[0].Message + " (" +
                                ex.Errors[0].NativeError.ToString() + ")",
                                "Failed to connect" );
                }
                catch(Exception ex)
                {
                        throw ex;
                }
                finally
                {
                        asa_conn.Close();
                }
        }
        #endregion
public DataRelation SetRelation(string relName,DataColumn parentCol,DataColumn childCol)
        {
                DataRelation rel;
                rel = new DataRelation(relName,parentCol,childCol);
                return rel;
        }
```

```
#region Создаём объекты Command для вставки,обновления и удаления в соответствующие им Adapter.Comm
        public AsaCommand CreateUpdateViaSp()
        {
            AsaCommand cmd = new AsaCommand("upd_curc",asa_conn);
            cmd.CommandType = CommandType.StoredProcedure;
            AsaParameterCollection pc = cmd.Parameters;
            AsaParameter param;
            param = pc.Add("incurc_id",AsaDbType.Integer,4,"curc_id");
            param.SourceVersion = DataRowVersion.Original;
            param.Direction = ParameterDirection.Input;
            pc.Add("incurc",AsaDbType.Char,3,"curc");
            pc.Add("insumma",AsaDbType.Char,17,"summa");
            pc.Add("inComments",AsaDbType.VarChar,50,"Comments");
            pc.Add("inbic",AsaDbType.Char,11,"bic");
            return cmd;
        }
        public AsaCommand CreateInsertViaSp()
        {
            AsaCommand cmd = new AsaCommand("add_curc",asa_conn);
            cmd.CommandType = CommandType.StoredProcedure;
            AsaParameterCollection pc = cmd.Parameters;
            pc.Add("incurc",AsaDbType.Char,3,"curc");
            pc.Add("insumma",AsaDbType.Char,17,"summa");
            pc.Add("inComments",AsaDbType.VarChar,50,"Comments");
            pc.Add("inbic",AsaDbType.Char,11,"bic");
            return cmd;
        }
        public AsaCommand CreateDeletetViaSp()
        {
            AsaCommand cmd = new AsaCommand("del_curc",asa_conn);
            cmd.CommandType = CommandType.StoredProcedure;
            AsaParameterCollection pc = cmd.Parameters;
            AsaParameter param;
            param = pc.Add("incurc_id",AsaDbType.Integer,4,"curc_id");
            param.SourceVersion = DataRowVersion.Original;
            param.Direction = ParameterDirection.Input;
        return cmd;
        }
        #endregion
        public int SubmitChangesViaSP(DataSet inds)
        {
            asa_adapter.UpdateCommand = CreateUpdateViaSp();
            asa_adapter.InsertCommand = CreateInsertViaSp();
            asa_adapter.DeleteCommand = CreateDeletetViaSp();
            int intCurcModified;
            intCurcModified = -1;
            if(inds.HasChanges())
            {
                try
                {
                    intCurcModified = asa_adapter.Update(inds.Tables[2]);
```

```
                                        string strOutput;
                                        strOutput = "Modified " + intCurcModified + " curc(s)";
        MessageBox.Show(strOutput,"Update succeeded!",
        MessageBoxButtons.OK,MessageBoxIcon.Information);
                                }
                                catch(Exception ex)
                                {
                                        MessageBox.Show(ex.Message,"Update failed!",
                                MessageBoxButtons.OK,MessageBoxIcon.Error);
                                }
                        }
                        else
                        {
                                MessageBox.Show("No changes to submit!","Submit changes",
                                        MessageBoxButtons.OK,MessageBoxIcon.Information);
                        }
                        return intCurcModified;
                }
                public StringCollection GetBicCorr()
                {
                        StringCollection str_mess;
                        str_mess = new StringCollection();
                        asa_conn = new AsaConnection(str_conn);
                        try
                        {
                                asa_conn.Open();
string str_cmd = "select bic,curc,summa from curc c,bic b where c.bic_id=b.bic_id and b.stat='C'";
                                asa_cmd = new AsaCommand(str_cmd,asa_conn);
                                asa_cmd.CommandType = CommandType.Text;
                                AsaDataReader dreader = asa_cmd.ExecuteReader();
                                while(dreader.Read())
                                {
        str_mess.Add(dreader[0]+"#curc#"+dreader[1]+"#sum#"+dreader[2]);
                                }
                        }
                        catch(AsaException asaex)
                        {
                                LogException(asaex);
                                str_mess[0] = String.Copy(asaex.ToString());
                        }
                        catch(Exception ex)
                        {
                                str_mess[0] = String.Copy(ex.ToString());
                                throw ex;
                        }
                        finally
                        {
                                asa_conn.Close();
                        }
                        return str_mess;
                }
                private void LogException(AsaException exc)
```

```
                {
                        EventLog el = new EventLog();
                        el.Source = "ComGenerator";
                        string strMessage;
strMessage = "Exception Number : " + exc.Errors.Count +exc.Message + " has ocurred";
                        el.WriteEntry(strMessage);
                        foreach(AsaError asaerr in exc.Errors)
                        {
                                strMessage = "Message" + asaerr.Message+
                                        "Source " + asaerr.Source;
                                el.WriteEntry(strMessage);
                        }
                }
        }
}
```

CDB

```
using System;
using System.Data;
using System.Data.Common;
using iAnywhere.Data.AsaClient;
using System.Windows.Forms;
using System.Diagnostics;
using System.Collections.Specialized;
namespace ComGenerator
{
        /// <summary>
        /// Summary description for CDB.
        /// </summary>
        public class CDB
        {
                private string str_conn;
                private string str_cmd;
                private string[] str_rslt;
                //Anywhere data types
                private AsaCommand asa_cmd;
                private AsaConnection asa_conn;
                private AsaDataAdapter asa_adapter;
                //Common data types
                private DataSet ds;
                private DataTable dt;
                public CDB()
                {
                        str_conn = "Data Source=CHRGAS;UID=DBA;PWD=DBA";
                                        str_rslt = new string[500];
                        }
                public CDB(string conn,string cmd)
                {
                        str_conn = "Data Source=CHRGAS;UID=DBA;PWD=DBA";
                        str_conn = String.Copy(conn);
                        str_cmd = String.Copy(cmd);
                        str_rslt = new string[500];
```

```
                        }
        public CDB(string cmd)
        {
                        str_conn = "Data Source=CHRGAS;UID=DBA;PWD=DBA";
                str_cmd = String.Copy(cmd);
                str_rslt = new string[500];
                        }
        public bool SetConnection()
        {
                bool flag=false;
                try
                {
                        asa_conn = new AsaConnection(str_conn);
                        asa_conn.Open();
                        flag = true;
                }
                catch(AsaException ex )
                {
                        MessageBox.Show( ex.Errors[0].Source + " : "
                                + ex.Errors[0].Message + " (" +
                                ex.Errors[0].NativeError.ToString() + ")",
                                "Failed to connect" );
                }
                return flag;
        }
        public string[] GetUsData(string cmd)
        {
                asa_cmd = new AsaCommand();
                asa_cmd.Connection = asa_conn;
                asa_cmd.CommandText = cmd;
                AsaDataReader reader = asa_cmd.ExecuteReader();
                int i = 0;
                while(reader.Read())
                {
                        str_rslt[i] =  reader.GetString(0);
                        i++;
                }
                return str_rslt;
        }
        public string[] GetUsData()
        {
                try
                {
                        asa_cmd = new AsaCommand();
                        asa_cmd.Connection = asa_conn;
                        asa_cmd.CommandText = str_cmd;
                        asa_adapter = new AsaDataAdapter(asa_cmd);
                        ds = new DataSet();
                        DataColumnMapping[] colMapArr;
colMapArr = new DataColumnMapping[]{new DataColumnMapping("bic_id","bic_id"),
  new DataColumnMapping("bic","bic"),  new
DataColumnMapping("stat","stat")};asa_adapter.TableMappings.AddRange(new
```

```
System.Data.Common.DataTableMapping[] {
        new System.Data.Common.DataTableMapping("Table", "bic", colMapArr)});
        asa_adapter.Fill(ds);
                        dt = ds.Tables["Bic"];
                        for(int curCol = 0;curCol<dt.Columns.Count;curCol++)
                        {
            str_rslt[0] = str_rslt[0] + dt.Columns[curCol].ColumnName.Trim() +"\t";
                        }
                        string str_tmp = new string(' ',500);
                        str_tmp = str_tmp.Trim();
                        for(int curRow=0;curRow<dt.Rows.Count;curRow++)
                        {
                                for(int curCol=0;curCol<dt.Columns.Count;curCol++)
                                {
                    str_tmp = str_tmp +  dt.Rows[curRow][curCol].ToString().Trim() + "\t";
                                }
                                str_rslt[curRow+1] = String.Copy(str_tmp);
                                str_tmp = str_tmp.Remove(0,str_tmp.Length);
                        }
                    }
                    catch(AsaException ex )
                    {
                        MessageBox.Show( ex.Errors[0].Source + " : "
                            + ex.Errors[0].Message + " (" +
                            ex.Errors[0].NativeError.ToString() + ")",
                            "Failed to connect" );
                    }
                    catch(Exception ex)
                    {
                        throw ex;
                    }
                    finally
                    {
                        asa_conn.Close();
                    }
                    return str_rslt;
                }
                public DataSet GetBicDataSet()
                {
                    try
                    {
                        asa_cmd = new AsaCommand();
                        asa_cmd.Connection = asa_conn;
                        asa_cmd.CommandText ="Select bic_id,bic,stat from bic";
                        asa_adapter = new AsaDataAdapter(asa_cmd);
                        ds = new DataSet();
                        asa_adapter.TableMappings.Add("Table","bic");
                        DataColumnMapping[] colMapArr;
colMapArr = new DataColumnMapping[]{new DataColumnMapping("bic_id","bic_id"),
  new DataColumnMapping("bic","bic"),  new DataColumnMapping("stat","stat")};
asa_adapter.TableMappings[0].ColumnMappings.AddRange(colMapArr);
                        asa_adapter.Fill(ds);
```

```
                }
                catch(AsaException ex )
                {
                        MessageBox.Show( ex.Errors[0].Source + " : "+ ex.Errors[0].Message
+ " (" +ex.Errors[0].NativeError.ToString() + ")",    "Failed to connect" );
                }
                catch(Exception ex)
                {
                        throw ex;
                }
                finally
                {
                        asa_conn.Close();
                }
                return ds;
        }
        public int SubmitChanges(DataSet inds)
        {
                asa_adapter.UpdateCommand = CreateDataAdapterUpdateCmd();
                asa_adapter.InsertCommand = CreateDataAdapterInsertCmd();
                asa_adapter.DeleteCommand = CreateDataAdapterDeleteCmd();
                int intBicModified;
                intBicModified = -1;
                if(inds.HasChanges())
                {
                        try
                        {
                                intBicModified = asa_adapter.Update(inds.Tables[0]);
                                string strOutput;
        strOutput = "Modified " + intBicModified + " bic(s)";
        MessageBox.Show(strOutput,"Update succeeded!",
        MessageBoxButtons.OK,MessageBoxIcon.Information);
                        }
                        catch(Exception ex)
                        {
                                MessageBox.Show(ex.Message,"Update failed!",
                                        MessageBoxButtons.OK,MessageBoxIcon.Error);
                        }
                }
                else
                {
                        MessageBox.Show("No changes to submit!","Submit changes",
                                MessageBoxButtons.OK,MessageBoxIcon.Information);
                }
                return intBicModified;
        }
        public int SubmitChangesViaSP(DataSet inds)
        {
                asa_adapter.UpdateCommand = CreateUpdateViaSp();
                asa_adapter.InsertCommand = CreateInsertViaSp();
                asa_adapter.DeleteCommand = CreateDeletetViaSp();
                int intBicModified;
```

```
                intBicModified = -1;
                if(inds.HasChanges())
                {
                        try
                        {
                                intBicModified = asa_adapter.Update(inds.Tables[0]);
                                string strOutput;
                                strOutput = "Modified " + intBicModified + " bic(s)";
        MessageBox.Show(strOutput,"Update succeeded!",
        MessageBoxButtons.OK,MessageBoxIcon.Information);
                        }
                        catch(Exception ex)
                        {
                                MessageBox.Show(ex.Message,"Update failed!",
                                        MessageBoxButtons.OK,MessageBoxIcon.Error);
                        }
                }
                else
                {
                        MessageBox.Show("No changes to submit!","Submit changes",
                                MessageBoxButtons.OK,MessageBoxIcon.Information);
                }
                return intBicModified;
        }
        public AsaCommand CreateDataAdapterUpdateCmd()
        {
                string strSQL;
strSQL = "UPDATE bic SET bic_id = ?, bic = ?, stat = ? WHERE ( bic_id = ? ) AND ( bic = ? )" +
" AND ( ( ? IS NULL AND stat IS NULL ) OR ( stat = ? ) )";
                AsaCommand cmd = new AsaCommand(strSQL,asa_conn);
                AsaParameterCollection pc = cmd.Parameters;
                pc.Add("bic_id_new",AsaDbType.Integer,4,"bic_id");
                pc.Add("bic_new",AsaDbType.Char,11,"bic");
                pc.Add("stat_new",AsaDbType.Char,1,"stat");
                AsaParameter param;
                param = pc.Add("bic_id_orig",AsaDbType.Integer,4,"bic_id");
                param.SourceVersion = DataRowVersion.Original;
                param.Direction = ParameterDirection.Input;
                param = pc.Add("bic_orig",AsaDbType.Char,11,"bic");
                param.SourceVersion = DataRowVersion.Original;
                param.Direction = ParameterDirection.Input;
                param = pc.Add("stat_orig",AsaDbType.Char,1,"stat");
                param.SourceVersion = DataRowVersion.Original;
                param.Direction = ParameterDirection.Input;
                param = pc.Add("stat_orig2",AsaDbType.Char,1,"stat");
                param.SourceVersion = DataRowVersion.Original;
                param.Direction = ParameterDirection.Input;
                return cmd;
        }
        public AsaCommand CreateDataAdapterInsertCmd()
        {
                string strSQL = "INSERT INTO bic( bic, stat ) VALUES( ?, ? )";
```

```
        AsaCommand cmd = new AsaCommand(strSQL,asa_conn);
        AsaParameterCollection pc = cmd.Parameters;
        pc.Add("bic_inc",AsaDbType.Char,11,"bic");
        pc.Add("stat_ins",AsaDbType.Char,1,"stat");
        return cmd;
}
public AsaCommand CreateDataAdapterDeleteCmd()
{
        string strSQL = "DELETE FROM bic WHERE ( bic_id = ? )";
        AsaCommand cmd = new AsaCommand(strSQL,asa_conn);
        AsaParameterCollection pc = cmd.Parameters;
        AsaParameter param;
        param = pc.Add("bic_id_del",AsaDbType.Integer,4,"bic_id");
        param.SourceVersion = DataRowVersion.Original;
        return cmd;
}
public AsaCommand CreateUpdateViaSp()
{
        AsaCommand cmd = new AsaCommand("upd_bic",asa_conn);
        cmd.CommandType = CommandType.StoredProcedure;
        AsaParameterCollection pc = cmd.Parameters;
        AsaParameter param;
        param = pc.Add("in_bic_id",AsaDbType.Integer,4,"bic_id");
        param.SourceVersion = DataRowVersion.Original;
        param.Direction = ParameterDirection.Input;

        pc.Add("inbic",AsaDbType.Char,11,"bic");
        pc.Add("instat",AsaDbType.Char,1,"stat");
        return cmd;
}
public AsaCommand CreateInsertViaSp()
{
        AsaCommand cmd = new AsaCommand("add_bic",asa_conn);
        cmd.CommandType = CommandType.StoredProcedure;
        AsaParameterCollection pc = cmd.Parameters;
        pc.Add("inbic",AsaDbType.Char,11,"bic");
        pc.Add("instat",AsaDbType.Char,1,"stat");
        return cmd;
}
public AsaCommand CreateDeletetViaSp()
{
        AsaCommand cmd = new AsaCommand("del_bic",asa_conn);
        cmd.CommandType = CommandType.StoredProcedure;
        AsaParameterCollection pc = cmd.Parameters;
        AsaParameter param;
        param = pc.Add("bic_orig",AsaDbType.Char,11,"bic");
        param.SourceVersion = DataRowVersion.Original;
        param.Direction = ParameterDirection.Input;
        return cmd;
}
public bool CloseCon()
{
```

```
                asa_conn.Close();
                return true;
        }
        public StringCollection GetBicWhere(string cmd)
        {
                StringCollection str_mess;
                str_mess = new StringCollection();
                string str_rslt = new string(' ',1500);
                str_rslt = str_rslt.Trim();
                asa_conn = new AsaConnection(str_conn);
                try
                {
                        asa_conn.Open();
                        string str_cmd = cmd;
                        asa_cmd = new AsaCommand(str_cmd,asa_conn);
                        asa_cmd.CommandType = CommandType.Text;
                        AsaDataReader dreader = asa_cmd.ExecuteReader();
                        while(dreader.Read())
                        {
                                str_rslt = String.Copy(dreader[0]+"#"+dreader[1]);
                                str_mess.Add(str_rslt);
                        }
                }
                catch(AsaException asaex)
                {
                        LogException(asaex);
                        str_mess[0] = String.Copy(asaex.ToString());
                }
                catch(Exception ex)
                {
                        str_mess[0] = String.Copy(ex.ToString());
                        throw ex;
                }
                finally
                {
                        asa_conn.Close();
                }
                return str_mess;
        }
        private void LogException(AsaException exc)
        {
                EventLog el = new EventLog();
                el.Source = "ComGenerator";
                string strMessage;
strMessage = "Exception Number : " + exc.Errors.Count +exc.Message + " has ocurred";
                el.WriteEntry(strMessage);
                foreach(AsaError asaerr in exc.Errors)
                {
                        strMessage = "Message" + asaerr.Message+
                                "Source " + asaerr.Source;
                        el.WriteEntry(strMessage);
                }
```

```
		}
	}
}


CUsers
using System;
using System.Data;
using System.Data.Common;
using System.Windows.Forms;
using iAnywhere.Data.AsaClient;
using System.Diagnostics;
using System.Collections.Specialized;
namespace ComGenerator
{
	public class CUsers
	{
		//Connection parameters
		private string str_conn;
		private string str_cmd;
		//Anywhere data types
		private AsaCommand asa_cmd;
		private AsaConnection asa_conn;
		private AsaDataAdapter asa_adapter;
		public CUsers()
		{
			str_conn = "Data Source=CHRGAS;UID=DBA;PWD=DBA";
		}
		public bool SetConnection()
		{
			bool flag=false;
			try
			{
				asa_conn = new AsaConnection(str_conn);
				asa_conn.Open();
				flag = true;
			}
			catch(AsaException ex )
			{
				MessageBox.Show( ex.Errors[0].Source + " : "
					+ ex.Errors[0].Message + " (" +
					ex.Errors[0].NativeError.ToString() + ")",
					"Failed to connect" );
			}
			return flag;
		}
		public void GetUserDataSet(DataSet ds)
		{
str_cmd = "select c.curc_id,c.curc,c.summa,c.comments, b.bic from curc c,bic b "+"where
c.bic_id=b.bic_id";
			try
			{
```

```
                        asa_cmd = new AsaCommand();
                        asa_cmd.Connection = asa_conn;
                        asa_cmd.CommandText =str_cmd;
                        asa_adapter = new AsaDataAdapter(asa_cmd);
                        asa_adapter.TableMappings.Add("Table","curc");
                        DataColumnMapping[] colMapArr;
                        colMapArr = new DataColumnMapping[]{new
DataColumnMapping("curc_id","curc_id"),  new DataColumnMapping("curc","curc"),     new
DataColumnMapping("summa","summa"),new DataColumnMapping("Comments","Comments"),
new DataColumnMapping("bic","bic")};
                asa_adapter.TableMappings[0].ColumnMappings.AddRange(colMapArr);
                        asa_adapter.Fill(ds);
                    }
                    catch(AsaException ex )
                    {
                        MessageBox.Show( ex.Errors[0].Source + " : "
                            + ex.Errors[0].Message + " (" +
                            ex.Errors[0].NativeError.ToString() + ")",
                            "Failed to connect" );
                    }
                    catch(Exception ex)
                    {
                        throw ex;
                    }
                    finally
                    {
                        asa_conn.Close();
                    }
                }
        }
}
```

Модуль для создания комиссий

```
using System;
using System.Text;
using System.Collections.Specialized;
namespace ComGenerator
{
        /// <summary>
        /// //Генерируем MT191
        /// </summary>
        public class CGetMt191
        {
                private StringCollection str_allFrom103;
                private StringCollection str_bicDeny;
                private StringCollection str_52A;
                private StringCollection str_AllCorr;
                private string str_1600 = "1600";
                private string str_bic = new string(' ',11);
                private string str_f20 = new string(' ',16);
                private string str_f32A = new string(' ',28);
                private string str_50 = new string(' ',98);
```

```
		private string str_sndr = new string(' ',8);
		private string str_53A = new string(' ',8);
		private string str_54A = new string(' ',8);
		private string str_57A = new string(' ',8);
		private StringCollection mt_191 = FillShablon();
		private StringCollection mt191_mess;
		private StringCollection mt103_mess;
		private enum alpha{A=10,B,C,D,E,F};
public CGetMt191(StringCollection str_allFrom103,StringCollection str_bicDeny,StringCollection
str_52A)
		{
			this.str_allFrom103 = str_allFrom103;
			this.str_bicDeny = str_bicDeny;
			this.str_52A = str_52A;
			this.str_AllCorr = str_AllCorr;
			str_sndr = "-1";
			str_53A = "-1";
			str_54A = "-1";
			str_57A = "-1";
		}
		//Создаём шаблон, всем шаблонам шаблон:)
		private static StringCollection FillShablon()
		{
			StringCollection str_mess = new StringCollection();
str_mess[0] = "_1:F01PBANUA2XXXXX0000000000}{2:I191XXXXN}{3:{108:MTI}{4:";
				str_mess[1] = ":20:CHRG";
				str_mess[2] = ":21:";
				str_mess[3] = ":32B:";
				str_mess[4] = ":52A:PBANUA2X";
				str_mess[5] = ":57A:";
				str_mess[6] = ":71B:/COMM/OUR EXPENSES";
				str_mess[7] = ":72:/REC/FOR YR PMNT VAL. ";
				str_mess[8] = "//FOR ";
				str_mess[9] = "//B/O ";
				str_mess[10] = "//ACCRDNG TO YR INSTRUCTIONS 'OUR'";
				str_mess[11] = "-}0;
			return str_mess;
		}
		private string vvnNum(int fileCounter)
		{
			string str_NumFile = IToA(fileCounter,16);
			if(str_NumFile.Length == 1)
				str_NumFile = str_NumFile.Insert(0,"00");
			else if (str_NumFile.Length == 2)
				str_NumFile = str_NumFile.Insert(0,"0");
			return str_NumFile;
		}
		public string IToA(int num,int bace)
		{

			int i = 0;
			int len = GetLen(num);
```

```
                string mas = new string(' ',len);
                mas+=mas.Trim();
                while((num%bace)!=num)
                {
                        if(num%bace >= 10)
                                mas+= Enum.GetName(typeof(alpha),num%bace);
                        else
                                mas += num%bace;
                        num = num/bace;
                        if(num<bace && num !=0)
                        {
                                mas += num;
                        }
                }
                mas = mas.Trim();
                char[] tmp_mas = new char[len];

                for(i=mas.Length-1;i>=0;i--)
                {
                        tmp_mas[(mas.Length-1)-i] = mas[i];
                }
                string str = new string(tmp_mas);
                return str;
        }
        private int GetLen(int num)
        {
                int kol=1;
                do
                {
                        kol++;
                        num /=10;
                }
                while((num/10) != 0);
                return kol;
        }
//Проверяем на принадлежность отправитель 103/банка бенефициара к списку запрещеннцх
        private void CheckDeny()
        {
                bool flag=false;
                int count=0;
                string str_tmp = new string(' ',600);
                str_tmp = str_tmp.Trim();
                string sndr = new string(' ',11);
                sndr = sndr.Trim();
                string str52A = new string(' ',11);
                str52A = str52A.Trim();
                string str53 = new string(' ',20);
                str53 = str53.Trim();
                for(int i = 0;i< str_allFrom103.Count;i++)
                {
                        str_tmp = String.Copy(str_allFrom103[i]);
                        if(str_tmp.IndexOf(":53") !=-1)
```

```
                                {
                                        str53 = str_tmp.Substring(str_tmp.IndexOf(":53")+5,10);
                                }
                                if(str53.IndexOf("1600")!=-1)
                                {
                                        flag = true;
                                }
                                sndr = str_tmp.Substring(str_tmp.LastIndexOf("{2:O103")+10,8);
                                if(str_tmp.LastIndexOf(":52A:")!=-1)
                                {
str52A = str_tmp.Substring(str_tmp.LastIndexOf(":52A:"),str_tmp.IndexOf(":59")-
str_tmp.LastIndexOf(":52A:"));
str52A = str52A.Substring(0,str52A.IndexOf(":"));
                                }
                                count=0;
                                while(!flag)
                                {
                                        str_tmp = String.Copy(str_bicDeny[i]);
                                        if(str_tmp.IndexOf(sndr)!=-1)          flag=true;
                                        if(str_bicDeny.Count != count)
                                                count++;
                                }
                                count=0;
                                while(!flag)
                                {
                                        str_tmp = String.Copy(str_52A[i]);
                                        if(str_tmp.IndexOf(str52A)!=-1)       flag=true;
                                        if(str_52A.Count != count)
                                                count++;
                                }
                                if(flag)
                                {
                                        str_allFrom103.RemoveAt(i);
                                }
                        }
                }
//Проверяем сумму в сообщениии 103 - должна быть не меньше 5* сумму наших корров для
данной валюты
                //Передаем установленного sendera и vald
                private bool CheckSumm(string sndr,string summ,string vald)
                {
                        bool flag=false;
                        string str_tmp = new string(' ',100);
                        str_tmp = str_tmp.Trim();
                        for(int i=0;i<str_AllCorr.Count;i++)
                        {
                                str_tmp = str_AllCorr[i];
                                if(str_tmp.IndexOf(sndr)!=-1 && str_tmp.IndexOf(vald)!=-1)
                                {
                                        double chkSumm = System.Convert.ToDouble(summ);
double corrSumm =
System.Convert.ToDouble(str_tmp.Substring(str_tmp.LastIndexOf("#summ#")));
```

```
                                        if(corrSumm<chkSumm*5)
                                                flag=true;
                                }
                        }
                        return flag;
                }
                //Устанавливаем банк корреспондент для поля 57A сообщения MT103
                private string SetInfoSndr()
                {
                        return str_sndr;
                }
                private void SetFields(string mess)
                {
                        str_sndr = mess.Substring(mess.LastIndexOf("{2:O103"+10,8));
                        str_f20 = mess.Substring(mess.LastIndexOf(":20:"),16);
                        if(str_f20.IndexOf("#")!=-1)
                                str_f20 = str_f20.Substring(0,str_f20.IndexOf("#"));
                        str_f32A = mess.Substring(mess.LastIndexOf(":32A:"),28);
                        if(str_f32A.IndexOf("#") !=-1)
                                str_f32A = str_f32A.Substring(0,str_f32A.IndexOf("#"));
                }
        }
}
```

Главный модуль программы mFmsggen.

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;
namespace ComGenerator
{
        /// <summary>
        /// Summary description for Form1.
        /// </summary>
        public class mfmsgen : System.Windows.Forms.Form
        {
                private System.Windows.Forms.MainMenu mainMenu1;
                private System.Windows.Forms.MenuItem mFile;
                private System.Windows.Forms.MenuItem mFileGen;
                private System.Windows.Forms.MenuItem mFileExit;
                private System.Windows.Forms.MenuItem mHelp;
                private System.Windows.Forms.MenuItem mHelpAbout;
                private System.Windows.Forms.MenuItem menuHelpInfo;
                private System.Windows.Forms.MenuItem mFileBD;
                private System.String[] str_rslt;
                private System.Windows.Forms.MenuItem mReports;
                private System.Windows.Forms.MenuItem menuItem1;
                private System.Windows.Forms.MenuItem mInvestigations;
                private System.Windows.Forms.MenuItem menuItem2;
                private System.Windows.Forms.MenuItem menuItem3;
```

```
enum alpha{A=10,B,C,D,E,F};
/// <summary>
/// Required designer variable.
/// </summary>
private System.ComponentModel.Container components = null;
public mfmsgen()
{
	InitializeComponent();

	//
	// TODO: Add any constructor code after InitializeComponent call
	//
}

/// <summary>
/// Clean up any resources being used.
/// </summary>
protected override void Dispose( bool disposing )
{
	if( disposing )
	{
		if (components != null)
		{
			components.Dispose();
		}
	}
	base.Dispose( disposing );
}

#region Windows Form Designer generated code
/// <summary>
/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
/// </summary>
private void InitializeComponent()
{
	this.mainMenu1 = new System.Windows.Forms.MainMenu();
	this.mFile = new System.Windows.Forms.MenuItem();
	this.mFileGen = new System.Windows.Forms.MenuItem();
	this.mFileBD = new System.Windows.Forms.MenuItem();
	this.mFileExit = new System.Windows.Forms.MenuItem();
	this.mHelp = new System.Windows.Forms.MenuItem();
	this.mHelpAbout = new System.Windows.Forms.MenuItem();
	this.menuHelpInfo = new System.Windows.Forms.MenuItem();
	this.mReports = new System.Windows.Forms.MenuItem();
	this.menuItem1 = new System.Windows.Forms.MenuItem();
	this.mInvestigations = new System.Windows.Forms.MenuItem();
	this.menuItem2 = new System.Windows.Forms.MenuItem();
	this.menuItem3 = new System.Windows.Forms.MenuItem();
	//
	// mainMenu1
```

```
            this.mainMenu1.MenuItems.AddRange(new System.Windows.Forms.MenuItem[] {
                                                    this.mFile,
                                                    this.mInvestigations,
                                                    this.mReports,
                                                    this.mHelp});
                    this.mFile.Index = 0;
                    this.mFile.MenuItems.AddRange(new System.Windows.Forms.MenuItem[] {
                                                        this.mFileGen,
                                                        this.mFileBD,
                                                        this.mFileExit});
                    this.mFile.Text = "&Files";
                                        this.mFileGen.Index = 0;
                    this.mFileGen.Text = "Generate request";
                    this.mFileGen.Click += new System.EventHandler(this.mFileGen_Click);
                    this.mFileBD.Index = 1;
                    this.mFileBD.Text = "BD Administrator";
                    this.mFileBD.Click += new System.EventHandler(this.mFileBD_Click);
                    this.mFileExit.Index = 2;
                    this.mFileExit.Text = "Exit";
                    this.mFileExit.Click += new System.EventHandler(this.mFileExit_Click);
                                        this.mHelp.Index = 3;
                    this.mHelp.MenuItems.AddRange(new System.Windows.Forms.MenuItem[]
{  this.mHelpAbout,    this.menuHelpInfo});
                    this.mHelp.Text = "&Help";
                    this.mHelpAbout.Index = 0;
                    this.mHelpAbout.Text = "About";
        this.mHelpAbout.Click += new System.EventHandler(this.mHelpAbout_Click);
                    this.menuHelpInfo.Index = 1;
                    this.menuHelpInfo.Text = "Information";
                    this.mReports.Index = 2;
                this.mReports.MenuItems.AddRange(new System.Windows.Forms.MenuItem[] {
                this.menuItem1, this.menuItem2});
                    this.mReports.Text = "Reports";
                    this.menuItem1.Index = 0;
                    this.menuItem1.Text = "File Print";
                    this.mInvestigations.Index = 1;
this.mInvestigations.MenuItems.AddRange(new System.Windows.Forms.MenuItem[] {
        this.menuItem3});
                    this.mInvestigations.Text = "Investigations";
                    this.menuItem2.Index = 1;
                    this.menuItem2.Text = "Generate";
                    this.menuItem3.Index = 0;
                    this.menuItem3.Text = "";
                    this.AutoScaleBaseSize = new System.Drawing.Size(5, 13);
                    this.ClientSize = new System.Drawing.Size(552, 325);
```

```
        this.Menu = this.mainMenu1;
        this.Name = "mfmsgen";
        this.Text = "Commisions";
        this.Load += new System.EventHandler(this.mfmsgen_Load);
}
#endregion
[STAThread]
static void Main()
{
        Application.Run(new mfmsgen());
}

private void mFileExit_Click(object sender, System.EventArgs e)
{
        Application.Exit();
}

private void mHelpAbout_Click(object sender, System.EventArgs e)
{
        fAbout dAbout = new fAbout();
        Point newP = new Point(100,100);
        dAbout.DesktopLocation = newP;
        dAbout.ShowDialog();
        dAbout.Dispose();
}
private void mFileGen_Click(object sender, System.EventArgs e)
{
        fGenerate dGen = new fGenerate();
        dGen.ShowDialog();
        dGen.Dispose();
}
private void mFileBD_Click(object sender, System.EventArgs e)
{
        fBDLoad dDBLoad = new fBDLoad();
        Point newP = new Point(100,100);
        dDBLoad.DesktopLocation = newP;
        dDBLoad.ShowDialog();
        dDBLoad.Dispose();
}
public string IToA(int num,int bace)
{

        int i = 0;
        int len = GetLen(num);
        string mas = new string(' ',len);
        mas+=mas.Trim();
        while((num%bace)!=num)
        {
                if(num%bace >= 10)
                        mas+= Enum.GetName(typeof(alpha),num%bace);
                else
                        mas += num%bace;
```

```
                num = num/bace;
                if(num<bace && num !=0)
                {
                        mas += num;
                }
            }
            mas = mas.Trim();
            char[] tmp_mas = new char[len];

            for(i=mas.Length-1;i>=0;i--)
            {
                tmp_mas[(mas.Length-1)-i] = mas[i];
            }
            string str = new string(tmp_mas);
            return str;
        }
        private int GetLen(int num)
        {
            int kol=1;
            do
            {
                kol++;
                num /=10;
            }
            while((num/10) != 0);
            return kol;
        }

        private void mfmsgen_Load(object sender, System.EventArgs e)
        {
            Autent fAutent = new Autent();
            fAutent.ShowDialog();
            fAutent.Dispose();
        }
        private void testMessage()
        {
            string message = "Main form loading";
            string caption = "Test";
            MessageBoxButtons buttons = MessageBoxButtons.YesNo;
            DialogResult result;
    result = MessageBox.Show(this, message, caption, buttons,
                MessageBoxIcon.Question, MessageBoxDefaultButton.Button1,
                MessageBoxOptions.RightAlign);
        if(result == DialogResult.Yes)
            {
        this.Close();

            }
        }
    }
}
```

ДОДАТОК Б

ВІДГУК КЕРІВНИКА ЕКОНОМІЧНОГО РОЗДІЛУ

ДОДАТОК В

ПЕРЕЛІК ФАЙЛІВ НА ДИСКУ

Ім’я файлу	**Опис**
Пояснювальні документи	
Диплом_ doc	Пояснювальна записка до кваліфікаційної роботи. Документ Word.
Диплом_.pdf	Пояснювальна записка до кваліфікаційної роботи в форматі PDF
Програма	
Program.rar	Архів. Містить коди програми і откомпільовану програму
Презентація	
Презентація_.ppt	Презентація кваліфікаційної роботи