# Програмне та технічне забезпечення

## **Засоби розробки**

При створенні програмного продукту середовищем розробки було обрано Visual Studio 2010 для програмування мовою C#, використовуючи плагін ReSharper.

Дане середовище дозволяє розробляти як консольні застосунки, так і застосунки з графічним інтерфейсом, в тому числі з підтримкою технології Windows Forms.

Visual Studio включає в себе редактор вихідного коду з підтримкою технології IntelliSense і можливістю найпростішого рефакторингу коду. Вбудовані інструменти включають в себе редактор форм для спрощення створення графічного інтерфейсу програми, веб-редактор, дизайнер класів і дизайнер схеми бази даних. Visual Studio дозволяє створювати і підключати сторонні доповнення (плагіни) для розширення функціональності практично на кожному рівні, включаючи додавання підтримки систем контролю версій вихідного коду, додавання нових наборів інструментів.

Плагін ReSharper дуже зручно використовувати при розробці мовою C#. Він проводить статичний аналіз коду (пошук помилок в коді до компіляції) в масштабі всього рішення, передбачає додаткові засоби автозаповнення, навігації, пошуку, підсвічування синтаксису, форматування, оптимізації та генерації коду, надає 40 автоматизованих рефакторингів.

C# – сучасна мова програмування, яка має дуже багато переваг перед такими мовами, як C/C++ або Java. Ось неповний їх список:

* C# створювався паралельно з платформою .NET Framework і повною мірою враховує всі його можливості;
* C# є повністю об'єктно-орієнтованою мовою, де навіть типи, вбудовані в мову, представлені класами;
* C# є потужною об'єктною мовою з можливостями наслідування і універсалізації;
* C# є спадкоємцем мов C/C++, зберігаючи кращі риси цих популярних мов програмування. Спільний з цими мовами синтаксис, знайомі оператори мови полегшують перехід програмістів від С++ до C#;
* зберігши основні риси C++, мова стала простіше і надійніше. Простота і надійність, головним чином, пов'язані з тим, що на C# хоча і допускаються, але не заохочуються такі небезпечні властивості С++ як покажчики, адресація, розіменування, адресна арифметика;
* завдяки платформі .NET Framework, що стала надбудовою над операційною системою, програмісти C # отримують ті ж переваги роботи з віртуальною машиною, що і програмісти Java. Ефективність коду навіть підвищується, оскільки виконавча середа CLR є компілятором проміжної мови, в той час як віртуальна Java-машина є інтерпретатором байт-коду;
* потужна бібліотека платформи підтримує зручність побудови різних типів застосунків на C#, дозволяючи легко будувати Web-служби, інші види компонентів, достатньо просто зберігати і отримувати інформацію з бази даних.

Для збереження інформації було обрано систему управління реляційними базами даних Microsoft SQL Server 2008 R2.

Це сучасна відмовостійка СУБД, яку дуже зручно використовувати при розробці застосунків на основі платформи .NET Framework. Платформа надає спеціальні засоби для роботи з Microsoft SQL Server.

Ця СУБД використовується для роботи з базами даних розміром від персональних до великих баз даних масштабу підприємства.

В якості системи контролю версій було використано розподілену систему керування версіями файлів Git.

Використання розподіленої системи контролю версій дає наступні переваги (над централізованими):

* періодична синхронізація декількох комп'ютерів під управлінням одного розробника (робочого комп'ютера, домашнього комп'ютера, ноутбука і так далі). Використання розподіленої системи позбавляє від необхідності виділяти один з комп'ютерів в якості сервера, а синхронізація виконується за потребою;
* спільна робота над проектом невеликою територіально розподіленою групи розробників без виділення загальних ресурсів. Як і в попередньому випадку, реалізується схема роботи без головного сервера, а актуальність репозиторіїв підтримується періодичними синхронізації за схемою «кожен з кожним».
* великий розподілений проект, учасники якого можуть довгий час працювати кожен над своєю частиною, при цьому не мають постійного підключення до мережі. Такий проект може використовувати централізований сервер, з яким синхронізуються копії всіх його учасників. Можливі й більш складні варіанти - наприклад, зі створенням груп для роботи за окремими напрямками всередині більшого проекту. При цьому можуть бути виділені окремі «групові» сервери для синхронізації роботи груп, тоді процес остаточного злиття змін стає деревовидним: спочатку окремі розробники синхронізують зміни на групових серверах, потім оновлені репозиторії груп синхронізуються з головним сервером. Можлива робота і без «групових» серверів, тоді розробники однієї групи синхронізують зміни між собою, після чого будь-який з них (наприклад, керівник групи) передає зміни на центральний сервер.

Використання Git дає наступні переваги (над аналогічними розподіленими системами контролю версій):

* висока продуктивність;
* продумана система команд;
* репозиторії git можуть поширюватися і оновлюватися загальносистемними файловими утилітами резервного копіювання та оновлення, завдяки тому, що фіксації змін і синхронізації не змінюють існуючі файли з даними, а лише додають нові (за винятком деяких службових файлів, які можуть бути автоматично оновлені за допомогою наявних у складі системи утиліт);
* для поширення репозиторію по мережі досить будь-якого веб-сервера.

## **Вимоги до технічного забезпечення**

### **Загальні вимоги**

Даний програмний продукт являє собою комплекс основних функцій процесу складання та прогнозування плану випуску продукції і призначений для використання співробітником підприємства, що працює у відповідному напрямі.

Для правильної роботи даної програми до складу технічних засобів повинні входити:

1. комп’ютер з такою конфігурацією:

1) процесор з тактовою частотою не нижче 1 ГГц;

1. достатній об’єм оперативної пам’яті (не менше 256 МБ);
2. інші складові можуть мати будь-які параметри, тому що вони не значним чином впливають на роботу програми;
3. додатково має бути встановлене таке програмне забезпечення:
4. операційна система Windows XP/Vista/Seven;
5. база даних Oracle Database 10g Express Edition;
6. Net Framework 3.5 і вище;
7. Microsoft Word 2003/2007/2010.
8. комп’ютерна периферія, до складу якої входить:
9. монітор;
10. мишка;
11. клавіатура;
12. принтер.

### **Опис локальної обчислювальної мережі**

*Описувати тільки у випадку коли комплекс задач:*

* *має трирівневу архітектуру;*
* *потребує використання локальної мережі.*

Мережа підприємств (рисунок 4.1) складається з ПЕОМ головного менеджера, ПЕОМ менеджерів відділу збуту, ПЕОМ менеджерів відділу планування виконання замовлень, ПЕОМ головного бухгалтера, ПЕОМ IT-спеціалістів.

ПЕОМ з’єднані в мережу за допомогою сітьового комутатора. Локальна мережа повинна бути захищена від зовнішніх чинників за допомогою міжмережевого екрану.

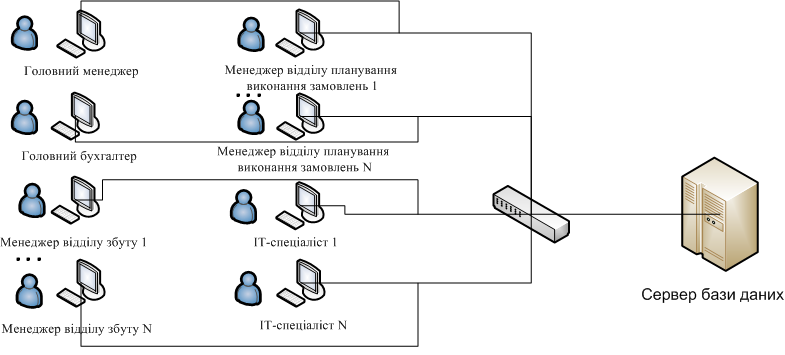


Рисунок 4.1 – Структура мережі

## **Архітектура програмного забезпечення**

### **Діаграма класів**

На рисунку 4.1 представлена структурна схема класів, які відповідають за виконання таких функцій програми, як встановлення з’єднання з СУБД Oracle, робота з даними, робота з транзакціями, складання та прогнозування плану випуску продукції та формування звітів «План виробництва».

Діаграма містить п’ять класів, а саме:

* «MainForm» - головний клас програми, відповідає за взаємодію програми з базою даних та містить основні функції роботи програми;
* «PasswordForm» - клас, що відповідає за коректне з’єднання з базою даних та запускає головний модуль програм;
* «ParametrPrograming» - клас у якому містяться функції, що виконують складання та прогнозування плану випуску продукції кожного виду;
* «ForecastPrice» - клас функції якого виконують прогнозування ринкових цін;
* «CreateReport» - клас призначенням якого є генерація звітів.

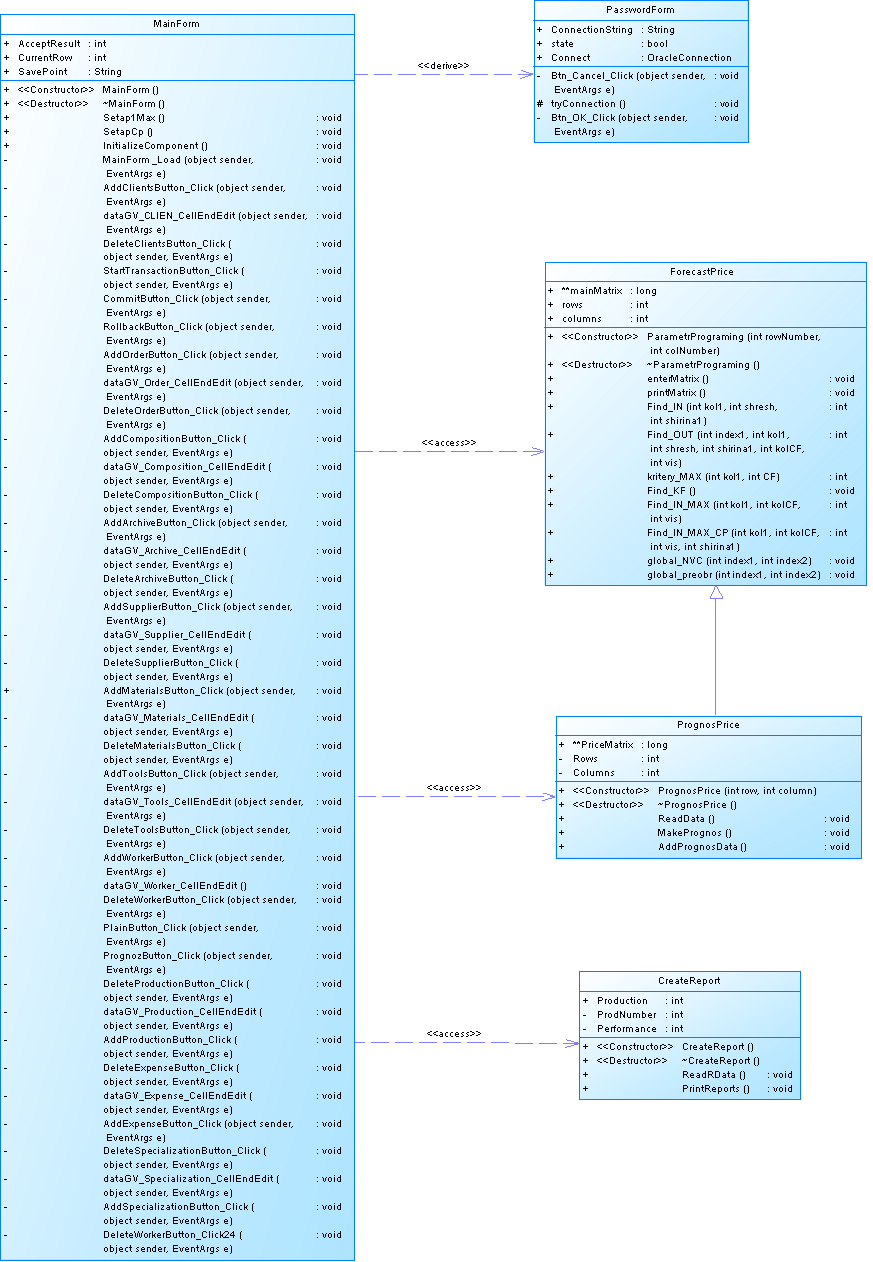


Рисунок ‎4.1 – Структурна схема класів

### **Діаграма послідовності**

На рисунку 4.2 представлена схема структурна послідовності. На даній діаграмі представлена типова послідовність дій, які виконуються при складенні оптимального плану виготовлення продукції, надання підприємству прогнозів плану виготовлення продукції на майбутні періоди та формуванні звіту «План виробництва». Також визначено акторів та приналежність їм відповідних дій, необхідних для виконання поставленої задачі.

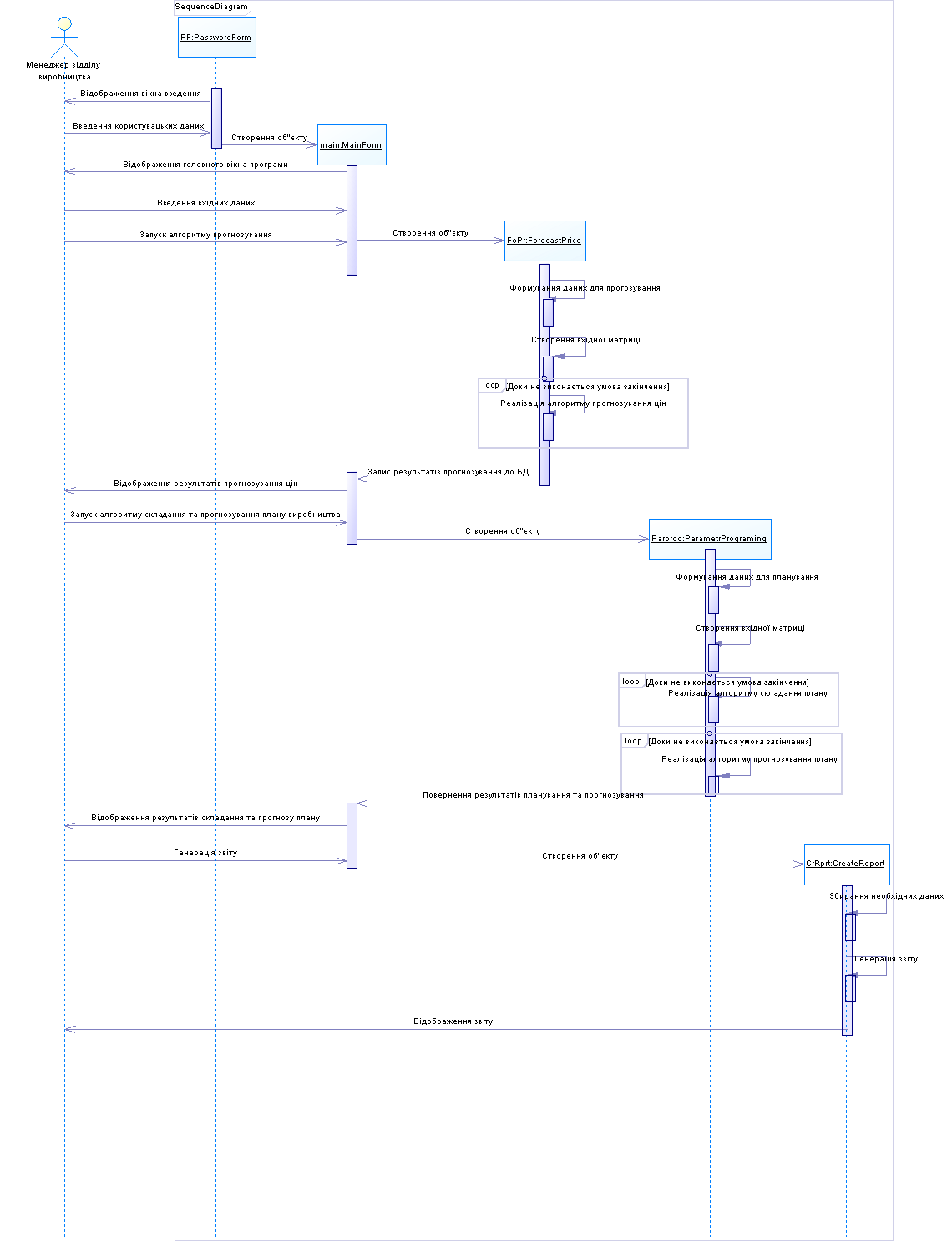


Рисунок ‎4.2 – Схема структурна послідовності

Далі, за допомогою діаграм послідовності опишемо дії, які виконуються при складанні плану випуску продукції та його прогнозуванні, а також при прогнозуванні ринкових цін.

На рисунку 4.3 представлена схема структурна послідовності дій роботи алгоритму. На даній діаграмі представлена типова послідовність дій, які виконуються при виконання алгоритму планування та прогнозування.



Рисунок ‎4.3 - Схема структурна послідовності дій роботи алгоритму планування та прогнозування

На рисунку 4.4 представлена схема структурна послідовності дій роботи алгоритму, який використовується при прогнозуванні ринкових цін за ретроспективними даними. На даній діаграмі представлена типова послідовність дій, які виконуються при виконання алгоритму прогнозування ринкових цін.

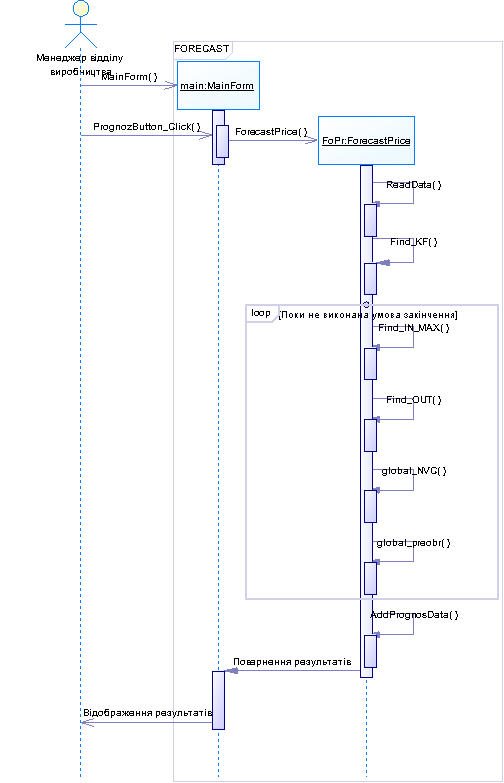


Рисунок ‎4.4 - Схема структурна послідовності дій роботи алгоритму прогнозування ринкових цін

### **Діаграма компонентів**

На рисунку 4.5 представлена схема структурна компонентів, на якій відображені компоненти, що використовуються в комплексі задач, та взаємозв’язки між ними. Основними компонентами в системі є співробітник підприємства, клієнтське застосування та СУБД Oracle.



Рисунок 4.5 – Схема структурна компонентів

### **Специфікація функцій**

*Приблизний приклад (таблиця наведена неповна)*

Функції класів програмного забезпечення наведені в таблиці 4.9.

Таблиця 4.9 – Функції класів програмного забезпечення

| **Клас** | **Функція** | **Опис функції** |
| --- | --- | --- |
| MainForm – головний клас, що відповідає за роботу з таблицями бази даних, містить інтерфейсні функції та викликає функції інших класів | public MainForm() | Реалізує запуск застосування. |
| public InitializeComponent() | Ініціалізує головний фрейм програми. |
| private void MainForm \_Load(object sender, EventArgs e) | Виконує завантаження наявних даних із БД у програму |
| public void InsertForcstResult() | Виконує запис до бази даних результатів прогнозування ринкових цін |
| public void InsertPlainData() | Виконує формування вхідних даних для планування |
| private void ExitButton\_Click(object sender, EventArgs e) | Виконує відключення від бази даних |
| private void AddClientsButton\_Click(object sender, EventArgs e) | Застосовується для додавання даних до таблиці клієнтів |
| private void dataGV\_CLIEN\_CellEndEdit(object sender, DataGridViewCellEventArgs e) | Застосовується для редагування даних в таблиці клієнтів |
| private void DeleteClientsButton\_Click(object sender, EventArgs e) | Застосовується для видалення даних з таблиці клієнтів |
| private void StartTransactionButton\_Click(object sender, EventArgs e) | Застосовується для початку транзакцій у базі даних |
| private void CommitButton\_Click(object sender, EventArgs e) | Застосовується для підтвердження здійснених змін у базі даних |
| private void RollbackButton\_Click(object sender, EventArgs e) | Застосовується для відміни останньої виконаної операції |
| private void AddOrderButton\_Click(object sender, EventArgs e) | Застосовується для додавання даних до таблиці договорів |

## **Опис звітів**

За допомогою програми можливий збір статистичних даних для побудови звітів «План виробництва».

Звіти створюються за допомогою відкритого COM-додатка, для роботи з сервером Microsoft Word. Даний додаток дозволяє посилати досить великий набір команд серверу Word і створювати звіти будь-якої складності та деталізації. При написанні коду будемо використовувати .Net збірку взаємодії з програмами Microsoft Office – Microsoft.Office.Interop.Word [12], для роботи нам потрібно лише одна частина даної збірки, а саме Word.Application. Головним в ієрархії об’єктів Word.Application є об’єкт Document. Інформація про об’єкти Document зберігається у вигляді посилань на відкриті документи у властивості Document. Document – це пустий лист, на якому генерується звіт «План виробництва».

Звіт «План виробництва» генерується автоматично при виборі в програмі пункту меню «Згенерувати звіт» і зберігається під назвою «План.doc» в папці Reports, що знаходиться в кореневій папці. Згенерований звіт має вигляд зображений на рисунку 4.6, де показано …

<Тут має бути зразок згенерованого звіту>

Рисунок 4.6 – Вигляд звіту «План виробництва»

## **Висновок до розділу**

*Дати висновок до цього розділу.*