Міністерство освіти і науки України

ВСП Технічний коледж

НУ «Львівська політехніка»

Звіт до лабораторної роботи №5-6

З предмету

Конструювання програмного забезпечення

Побудова за допомогою програми MS Visio UML діаграм автоматизованої інформаційної системи.

Виконала:

Студентка групи 43-ПЗ

Ткачук Анаствсія

Перевірив:

Ковалевич Т.С.

Львів 2016

**Лабораторна робота № 5-6**

Побудова за допомогою програми MS Visio UML діаграм автоматизованої інформаційної системи.

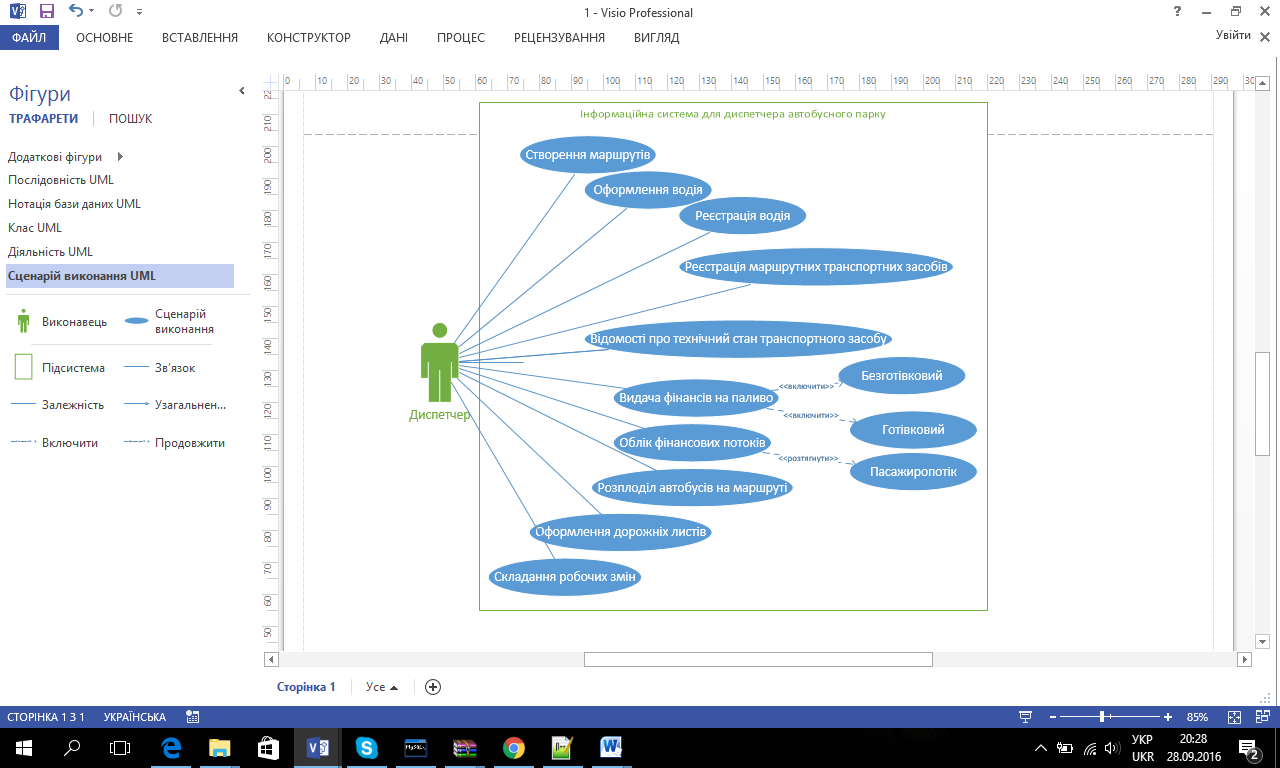
**Мета:** Засвоїти методику та виробити практичні навички роботи в редакторі діаграм і блок-схем Microsoft Visio та на його аналогах на основі створення UML діаграм.

**Частина 1**

**Завдання:** Розробити UML діаграми варіантів використання автоматизованої інформаційної системи згідно варіанту.

1. Інформаційна системи для диспетчера автобусного парку.

# Діаграма



**Частина 2**

**Завдання**

Побудувати за допомогою програми Rational Rose UML діаграму класів згідно варіанту.

**Варіант 1**

/\*

Super class - Page

\*/

class Page

{

private double width;

private double height;

//Constructor with no parameters

public Page()

{

width = 0;

height = 0;

}

//Constructor with paramaters

public Page(float width, float height)

{

this.width = width;

this.height = height;

}

//Destructor

~Page()

{

Console.WriteLine("Destructor");

}

//Method setting width

public void setWidth(float width)

{

this.width = width;

}

//Method getting width

public double getWidth()

{

return width;

}

//Method setting height

public void setHeight(float height)

{

this.height = height;

}

//Method getting height

public double getHeight()

{

return height;

}

//Static function

public static void staticFunction()

{

Console.WriteLine("Static function was called");

}

}

/\*

Class - Image

\*/

class Image

{

private float width;

private float height;

private String source;

//Constructor with paramaters

public Image(float width, float height, String source)

{

this.width = width;

this.height = height;

this.source = source;

}

//Method setting source

public void setSource(String source)

{

this.source = source;

}

//Method getting source

public String getSource()

{

return source;

}

}

/\*

Sub class - Main

\*/

class Main:Page

{

private static int count = 0;

private Image image = new Image(250, 250, "MainImage.jpg");

private String title;

private String text;

//Constructor with parameters

public Main(String title, String text)

{

count++;

this.title = title;

this.text = text;

Console.WriteLine("Creating "+ count + " Main Page");

}

//Method for changing image source

public void changeImageSource(String source)

{

image.setSource(source);

}

//Method getting source

public void getImageSource()

{

Console.WriteLine("Image source is: "+ image.getSource());

}

//Method getting count

public int getCount()

{

return count;

}

}

/\*

Sub class - About

\*/

class About : Page

{

private static int count = 0;

private Image image = new Image(250, 250, "AboutImage.jpg");

private String title;

private String text;

//Constructor with parameters

public About(String title, String text)

{

count++;

this.title = title;

this.text = text;

Console.WriteLine("Creating " + count + " About Page");

}

//Method for changing image source

public void changeImageSource(String source)

{

image.setSource(source);

}

//Method getting source

public void getImageSource()

{

Console.WriteLine("Image source is: " + image.getSource());

}

//Method getting count

public int getCount()

{

return count;

}

}

/\*

Sub class - News

\*/

class News : Page

{

private static int count = 0;

private Image image = new Image(250, 250, "NewsImage.jpg");

private String title;

private String text;

//Constructor with parameters

public News(String title, String text)

{

count++;

this.title = title;

this.text = text;

Console.WriteLine("Creating " + count + " News Page");

}

//Method for changing image source

public void changeImageSource(String source)

{

image.setSource(source);

}

//Method getting source

public void getImageSource()

{

Console.WriteLine("Image source is: " + image.getSource());

}

//Method getting count

public int getCount()

{

return count;

}

}

class Program

{

static void Main(string[] args)

{

//Caling static function

Page.staticFunction();

Console.WriteLine();

//Creating first Main page and changing all possible parameters

Main main1 = new Main("Hello", "This text references to Hello Main Page");

//Changing parameters

main1.changeImageSource("Hello.jpg");

main1.setHeight(400);

main1.setWidth(400);

//Displaying parameters

main1.getImageSource();

Console.WriteLine("Image width is: " + main1.getWidth());

Console.WriteLine("Image height is: " + main1.getHeight());

Console.WriteLine();

//Creating second Main page but leaving all parameters default

Main main2 = new Main("Goodbye", "This text references to Goodbye Main Page");

//Displaying parameters

main2.getImageSource();

Console.WriteLine("Image width is: " + main2.getWidth());

Console.WriteLine("Image heightf is: " + main2.getHeight());

Console.WriteLine();

//Displaying how many Main pages are created

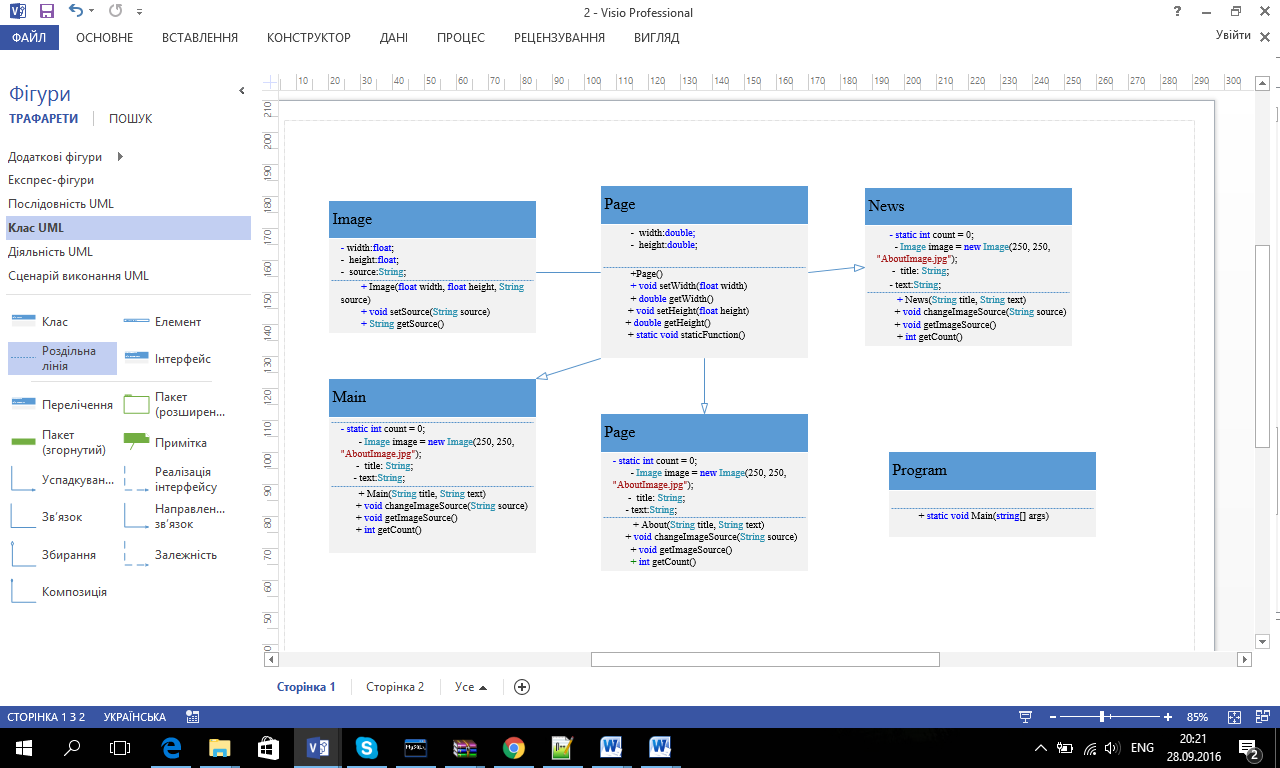
Console.WriteLine("There are "+main1.getCount()+" Main pages created");

Console.ReadKey();

}

}

**Діаграма**



**Частина 3**

Завдання

Побудувати за допомогою програми Rational Rose UML діаграму діяльності згідно варіанту.

Утворити масив i вивести на екран елементи масиву. *xi =* arctan(*i*ln*i)*. Обчислити, *де 1<=i<=10, крок=1.*

#include<stdio.h>

#include<conio.h>

#include<math.h>

main()

{

int i;

float x[10];

float S, t;

clrscr();

printf("Utvorenuy masuv\n");

for(i=0;i<10;i++)

{

x[i]=atan(i\*log(i));

printf("x[%d]= %lf\n", i, x[i]);

}

S=0;

for(i=0;i<10;i++)

{

S=S+pow(x[i],2);

}

t=sqrt(S);

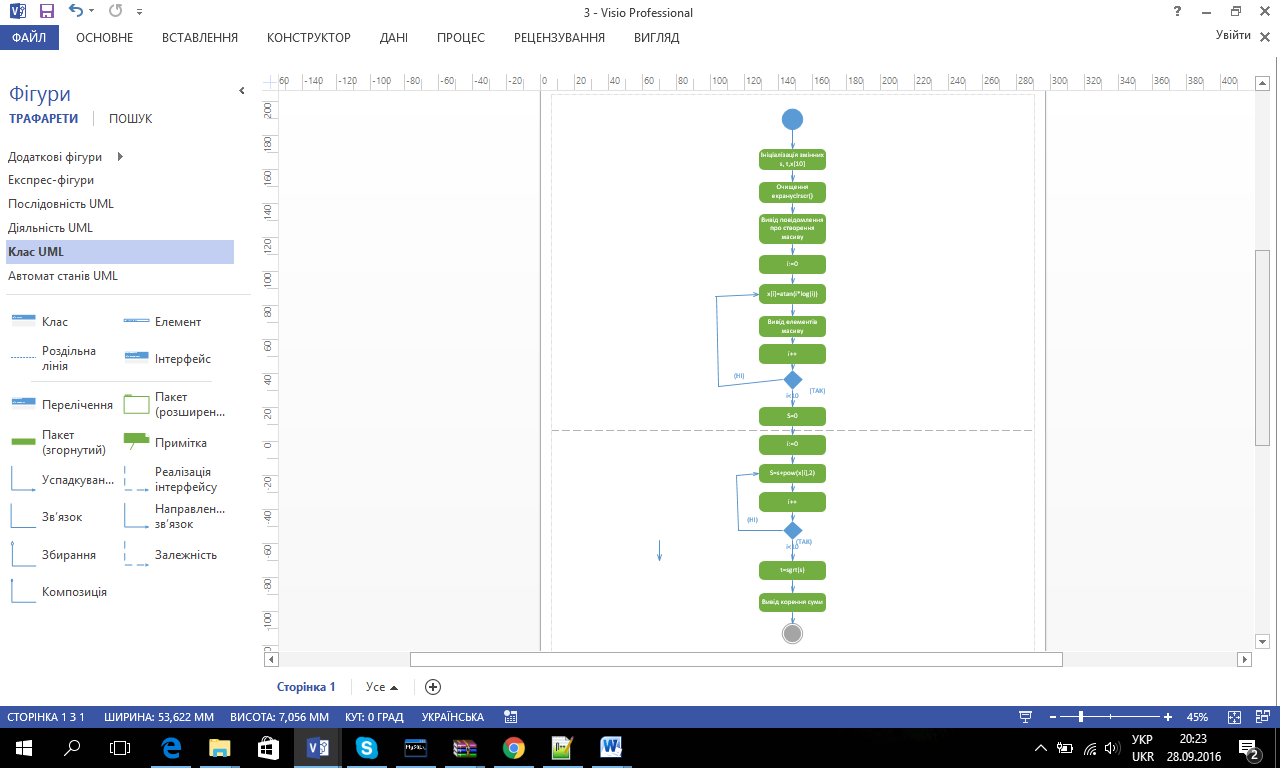
printf("Korin iz sumu = %lf", t);

getch();

return 0;

}

# Діаграма



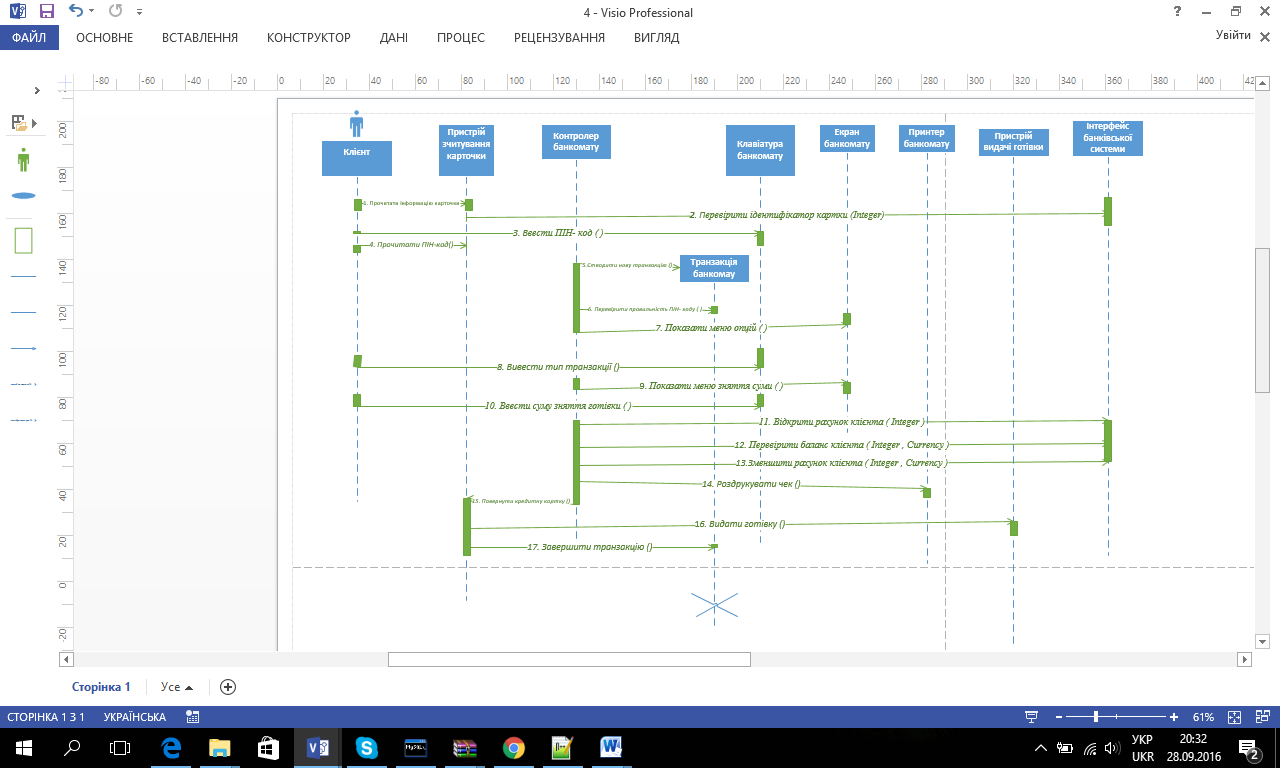
**Частина 4**

Хід лабораторної роботи

Для завершення побудови діаграми послідовності взаємодії користувача з банкоматом потрібно додати об'єкти і повідомлення , що залишилися. З цією метою слід виконати наступні дії:

1. Додати об'єкти класів з іменами: *Контролер Банкомату, Транзакція Банкомату, Клавіатура Банкомату , Екран Банкомату, Принтер Банкомату, Пристрій видачі готівки і Інтерфейс Банківської системи*.
2. Додати повідомлення : *перевірити ідентифікатор картки (Integer)* , спрямоване від об'єкта класу *Контролер Банкомату* до об'єкта класу *Інтерфейс Банківської системи*.
3. Додати повідомлення : *ввести ПІН- код ( )* , спрямоване від об'єкта класу - актора *Клієнт Банкомату* до об'єкта класу *Клавіатура Банкомати*.
4. Додати повідомлення : *прочитати ПІН- код ( )* , спрямоване від об'єкта класу *Контролер Банкомату* до об'єкта класу *Пристрій читання картки* .
5. Додати повідомлення : *створити нову транзакцію ( )* , спрямоване від об'єкта класу *Контролер Банкомату* до зображення об'єкта класу *Транзакція Банкомати*. При цьому зображення об'єкта класу *Транзакція Банкомату* слід перемістити вниз на рівень цього повідомлення , що буде візуально означатиме створення даного об'єкта в більш пізній момент часу , ніж початок функціонування модельованої програмної системи .
6. Додати повідомлення : *перевірити правильність ПІН- коду ( )* , спрямоване від об'єкта класу *Контролер Банкомату* до об'єкта класу *Транзакція Банкомати*.
7. Додати повідомлення : показати меню опцій ( ) , спрямоване від об'єкта класу *Контролер Банкомату* до об'єкта класу *Екран Банкомати*.
8. Додати повідомлення : *ввести тип транзакції ( )* , спрямоване від об'єкта класу - актора *Клієнт Банкомату* до об'єкта класу *Клавіатура Банкомати*.
9. Додати повідомлення : *показати меню зняття суми ( )* , спрямоване від об'єкта класу *Контролер* *Банкомату* до об'єкта класу *Екран Банкомати*.
10. Додати повідомлення *: ввести суму зняття готівки ( )* , спрямоване від об'єкта класу - актора *Клієнт Банкомату* до об'єкта класу *Клавіатура Банкомату*.
11. Послідовно додати 3 повідомлення: *відкрити рахунок клієнта ( Integer )* , *перевірити баланс клієнта ( Integer , Currency* ) *і зменшити рахунок клієнта ( Integer , Currency )*, спрямовані від об'єкта класу *Контролер Банкомату* до об'єкта класу *Інтерфейс Банківської системи*.
12. Додати повідомлення : *роздрукувати чек ( )* , спрямоване від об'єкта класу *Контролер Банкомату* до об'єкта класу *Принтер Банкомати*.
13. Додати повідомлення : *повернути кредитну картку ( )* , спрямоване від об'єкта класу *Контролер Банкомату* до об'єкта класу *Пристрій читання картки* .
14. Додати повідомлення : видати готівку ( ) , спрямоване від об'єкта класу *Контролер Банкомату* до об'єкта класу *Пристрій видачі готівки*.
15. Додати повідомлення : *завершити транзакцію ( )* , спрямоване від об'єкта класу *Контролер Банкомату* до об'єкта класу *Транзакція Банкомати*.
16. Після додавання повідомлення *завершити транзакцію (* ) помістити на лінію життя об'єкта класу Транзакція Банкомату символ знищення цього об'єкта.

# Діаграма



**Висновок**

Я засвоїла методику та виробила практичні навички роботи в редакторі діаграм і блок-схем Microsoft Visio та на його аналогах на основі створення UML діаграм.