**Технологии конструирования программного обеспечения**

**Отчет по лабораторной работе № 5**

**Группа:** 221-329 **Студент:** Ласурова Диана Эдуардовна

**Задание на лабораторную работу**

1 Изучить пример проектирования программной системы с использованием паттерна Состояние [Турчин-Архитектура ИС.pdf [Электронный ресурс], c. 143–154].

2 Разработать диаграмму конечных автоматов (состояний) для заданного класса (таблица 1). Описать в форме таблицы варианты реакции экземпляра класса на операции, вызываемые в указанных состояниях. Разработать UML-диаграммы классов и диаграммы последовательности.

3 Разработать библиотеку классов, включающую необходимые классы для реализации паттерна Состояние (класс Конечный автомат, интерфейс Состояние, классы Конкретные состояния).

4 : Разработать приложение Windows Forms для управления состояниями экземпляров класса Конечный автомат. При использовании Windows-форм вместо исходного кода в отчет вставить ссылку на репозиторий GitHub с проектом.

В-2

**Диаграмма конечных автоматов**

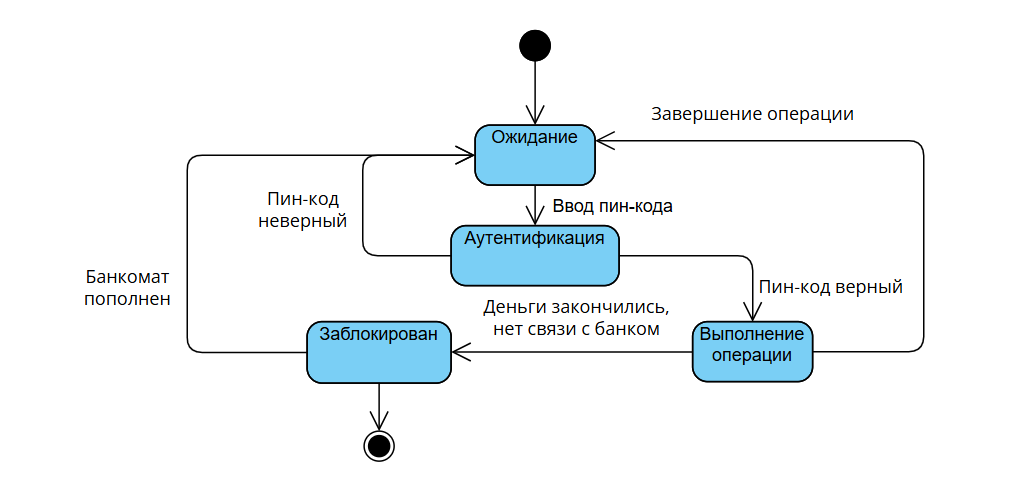
****

Рисунок 1. Диаграмма конечных автоматов

**Диаграмма классов**

Изображение выглядит как текст, линия, Шрифт, диаграмма

Автоматически созданное описание

Рисунок 2. Диаграмма классов

**Диаграмма последовательности**

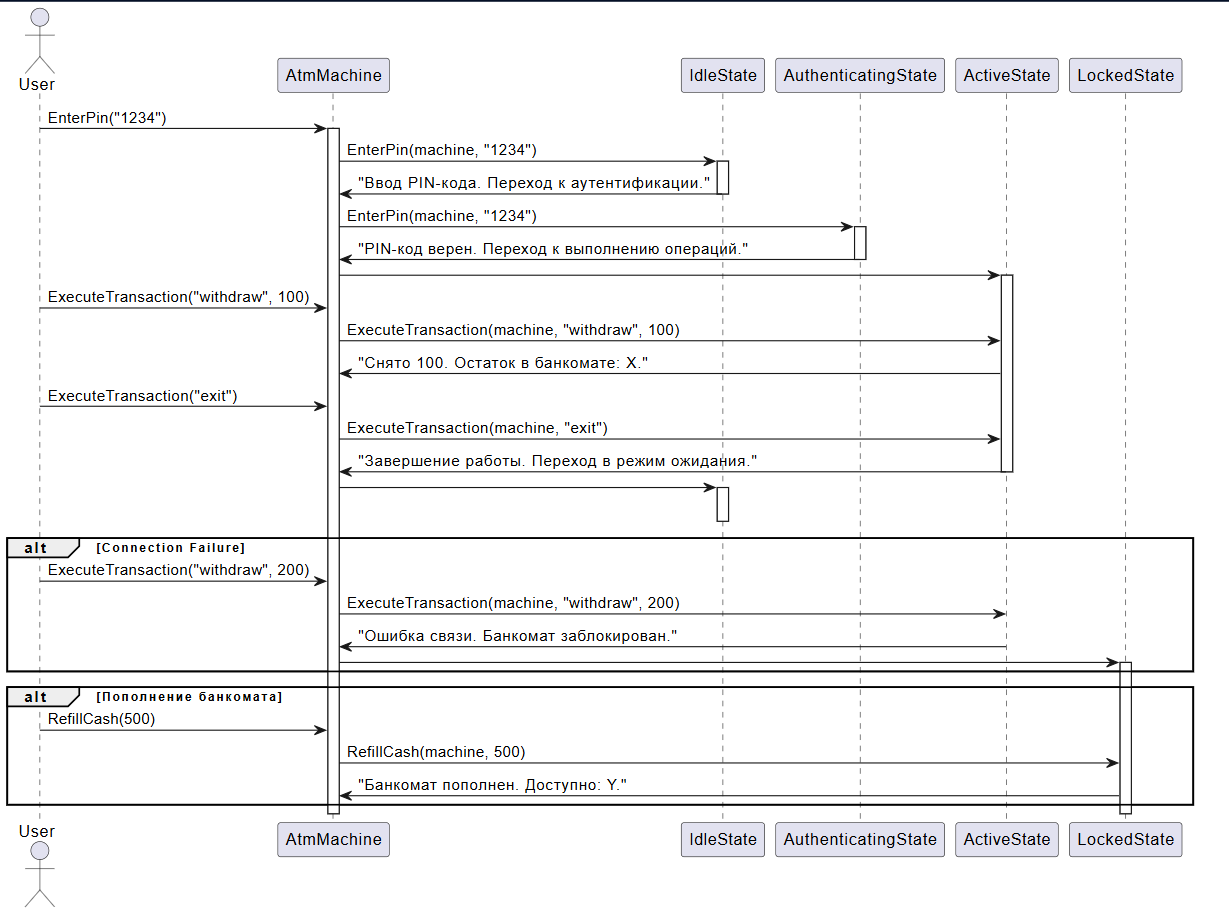
****

Рисунок 3. Диаграмма последовательности

**Исходный код программы**

using ClassLibrary; // Подключение пространства имен библиотеки классов, содержащей логику банкомата.

namespace ATMConsoleApp // Определение пространства имен консольного приложения для работы с банкоматом.

{

internal class Program // Объявление класса программы.

{

static void Main(string[] args) // Точка входа в приложение.

{

// Создаем экземпляр банкомата, указывая ID, начальный баланс, статус подключения и вероятность сбоя соединения.

var atm = new AtmMachine(1, 10000, true, 0.1);

atm.Alert = Console.WriteLine; // Настраиваем обработчик уведомлений для вывода сообщений на консоль.

Console.WriteLine("Добро пожаловать в банкомат!"); // Приветственное сообщение для пользователя.

bool isRunning = true; // Переменная для управления основным циклом работы программы.

while (isRunning) // Основной цикл программы, продолжается, пока переменная isRunning равна true.

{

// Вывод меню действий для пользователя.

Console.WriteLine("\nВыберите действие:");

Console.WriteLine("1. Ввести PIN-код");

Console.WriteLine("2. Снять деньги");

Console.WriteLine("3. Пополнить банкомат");

Console.WriteLine("4. Завершить работу");

Console.Write("Ваш выбор: ");

string choice = Console.ReadLine(); // Считываем выбор пользователя из консоли.

switch (choice) // Выполняем действие в зависимости от выбора пользователя.

{

case "1": // Если пользователь выбрал ввод PIN-кода.

Console.Write("Введите PIN-код: "); // Просим ввести PIN-код.

string pin = Console.ReadLine(); // Считываем PIN-код из ввода пользователя.

atm.EnterPin(pin); // Передаем PIN-код в метод банкомата для проверки.

break;

case "2": // Если пользователь выбрал снятие денег.

Console.Write("Введите сумму для снятия: "); // Просим ввести сумму для снятия.

if (int.TryParse(Console.ReadLine(), out int withdrawAmount)) // Проверяем корректность введенной суммы.

{

atm.ExecuteTransaction("withdraw", withdrawAmount); // Передаем запрос на снятие денег банкомату.

}

else // Если ввод некорректен.

{

Console.WriteLine("Ошибка: введите корректное значение суммы."); // Сообщаем об ошибке.

}

break;

case "3": // Если пользователь выбрал пополнение банкомата.

Console.Write("Введите сумму для пополнения: "); // Просим ввести сумму для пополнения.

if (int.TryParse(Console.ReadLine(), out int refillAmount)) // Проверяем корректность введенной суммы.

{

atm.RefillCash(refillAmount); // Передаем запрос на пополнение банкомата.

}

else // Если ввод некорректен.

{

Console.WriteLine("Ошибка: введите корректное значение суммы."); // Сообщаем об ошибке.

}

break;

case "4": // Если пользователь выбрал завершение работы.

atm.ExecuteTransaction("exit"); // Передаем запрос на завершение работы банкомату.

isRunning = false; // Завершаем основной цикл программы.

break;

default: // Если пользователь выбрал некорректный пункт меню.

Console.WriteLine("Неверный выбор. Пожалуйста, выберите действие из списка."); // Сообщаем об ошибке.

break;

}

}

Console.WriteLine("Спасибо за использование банкомата. До свидания!"); // Прощальное сообщение для пользователя.

}

}

}

using System; // Подключение пространства имен для стандартных классов C#, включая консольный ввод/вывод и делегаты.

namespace ClassLibrary // Определение пространства имен для логики банкомата.

{

// Определение интерфейса для состояний банкомата.

public interface IAtmState

{

// Метод для обработки ввода PIN-кода.

void EnterPin(AtmMachine machine, string pin);

// Метод для выполнения транзакций.

void ExecuteTransaction(AtmMachine machine, string transactionType, int amount = 0);

// Метод для пополнения наличных в банкомате.

void RefillCash(AtmMachine machine, int amount);

}

// Класс, представляющий банкомат.

public class AtmMachine

{

public int MachineId { get; set; } // Уникальный идентификатор банкомата.

public IAtmState CurrentState { get; set; } // Текущее состояние банкомата.

public int AvailableCash { get; set; } // Количество доступных наличных в банкомате.

public bool IsOnline { get; set; } // Статус соединения банкомата с сервером.

public double ConnectionFailureRate { get; set; } // Вероятность сбоя соединения.

private static Random random = new Random(); // Генератор случайных чисел для симуляции сбоев связи.

public Action<string> Alert { get; set; } // Делегат для уведомлений, принимающий строку.

// Конструктор для инициализации банкомата.

public AtmMachine(int machineId, int initialCash, bool isOnline, double connectionFailureRate)

{

MachineId = machineId; // Присваиваем ID.

AvailableCash = initialCash; // Устанавливаем начальный баланс наличных.

IsOnline = isOnline; // Устанавливаем статус соединения.

ConnectionFailureRate = connectionFailureRate; // Задаем вероятность сбоя соединения.

CurrentState = new IdleState(); // Устанавливаем начальное состояние (режим ожидания).

}

// Метод для передачи ввода PIN-кода в текущее состояние.

public void EnterPin(string pin) => CurrentState.EnterPin(this, pin);

// Метод для выполнения транзакций (например, снятие или выход).

public void ExecuteTransaction(string transactionType, int amount = 0)

{

SimulateConnectionIssue(); // Проверяем возможный сбой соединения.

CurrentState.ExecuteTransaction(this, transactionType, amount); // Передаем транзакцию текущему состоянию.

}

// Метод для пополнения наличных.

public void RefillCash(int amount) => CurrentState.RefillCash(this, amount);

// Метод для симуляции сбоев связи.

private void SimulateConnectionIssue()

{

if (random.NextDouble() < ConnectionFailureRate) // Если случайное число меньше вероятности сбоя.

{

IsOnline = false; // Отключаем соединение.

}

}

}

// Состояние ожидания (Idle).

public class IdleState : IAtmState

{

// Ввод PIN-кода из режима ожидания.

public void EnterPin(AtmMachine machine, string pin)

{

machine.Alert?.Invoke("Ввод PIN-кода. Переход к аутентификации.");

machine.CurrentState = new AuthenticatingState(); // Меняем состояние на "аутентификация".

machine.CurrentState.EnterPin(machine, pin); // Передаем PIN-код новому состоянию.

}

// Транзакции недоступны в режиме ожидания.

public void ExecuteTransaction(AtmMachine machine, string transactionType, int amount = 0)

{

machine.Alert?.Invoke("Пожалуйста, сначала введите PIN-код.");

}

// Пополнение недоступно в режиме ожидания.

public void RefillCash(AtmMachine machine, int amount)

{

machine.Alert?.Invoke("Невозможно пополнить банкомат в режиме ожидания.");

}

}

// Состояние аутентификации.

public class AuthenticatingState : IAtmState

{

// Проверка PIN-кода.

public void EnterPin(AtmMachine machine, string pin)

{

if (pin == "1234") // Если PIN-код правильный.

{

machine.Alert?.Invoke("PIN-код верен. Переход к выполнению операций.");

machine.CurrentState = new ActiveState(); // Переходим в активное состояние.

}

else // Если PIN-код неверный.

{

machine.Alert?.Invoke("Неверный PIN-код. Возврат в режим ожидания.");

machine.CurrentState = new IdleState(); // Возвращаемся в режим ожидания.

}

}

// Транзакции недоступны во время аутентификации.

public void ExecuteTransaction(AtmMachine machine, string transactionType, int amount = 0)

{

machine.Alert?.Invoke("Аутентификация ещё не завершена.");

}

// Пополнение недоступно во время аутентификации.

public void RefillCash(AtmMachine machine, int amount)

{

machine.Alert?.Invoke("Невозможно пополнить банкомат во время аутентификации.");

}

}

// Активное состояние.

public class ActiveState : IAtmState

{

// Повторный ввод PIN-кода в активном состоянии.

public void EnterPin(AtmMachine machine, string pin)

{

machine.Alert?.Invoke("Вы уже вошли в систему. Выполните операцию или завершите работу.");

}

// Выполнение транзакции.

public void ExecuteTransaction(AtmMachine machine, string transactionType, int amount = 0)

{

if (!machine.IsOnline) // Если банкомат не подключен.

{

machine.Alert?.Invoke("Ошибка связи. Банкомат заблокирован.");

machine.CurrentState = new LockedState(); // Переходим в заблокированное состояние.

return;

}

switch (transactionType.ToLower()) // Определяем тип транзакции.

{

case "withdraw": // Снятие наличных.

if (amount <= machine.AvailableCash && amount > 0) // Если достаточно средств.

{

machine.AvailableCash -= amount; // Снимаем сумму.

machine.Alert?.Invoke($"Снято {amount}. Остаток в банкомате: {machine.AvailableCash}.");

if (machine.AvailableCash <= 0) // Если деньги закончились.

{

machine.Alert?.Invoke("Деньги закончились. Банкомат заблокирован.");

machine.CurrentState = new LockedState(); // Переходим в заблокированное состояние.

}

}

else

{

machine.Alert?.Invoke("Недостаточно средств или неверная сумма.");

}

break;

case "exit": // Завершение работы.

machine.Alert?.Invoke("Завершение работы. Переход в режим ожидания.");

machine.CurrentState = new IdleState(); // Возвращаемся в режим ожидания.

break;

default: // Неизвестная операция.

machine.Alert?.Invoke("Неизвестная операция. Попробуйте снова.");

break;

}

}

// Пополнение недоступно в активном состоянии.

public void RefillCash(AtmMachine machine, int amount)

{

machine.Alert?.Invoke("Невозможно пополнить банкомат во время выполнения операций.");

}

}

// Заблокированное состояние.

public class LockedState : IAtmState

{

// Ввод PIN-кода невозможен.

public void EnterPin(AtmMachine machine, string pin)

{

machine.Alert?.Invoke("Банкомат заблокирован. Операции невозможны.");

}

// Выполнение транзакции невозможно.

public void ExecuteTransaction(AtmMachine machine, string transactionType, int amount = 0)

{

machine.Alert?.Invoke("Банкомат заблокирован. Операции невозможны.");

}

// Пополнение возможно и может разблокировать банкомат.

public void RefillCash(AtmMachine machine, int amount)

{

machine.AvailableCash += amount; // Пополняем баланс.

machine.Alert?.Invoke($"Банкомат пополнен. Доступно: {machine.AvailableCash}.");

if (machine.IsOnline && machine.AvailableCash > 0) // Если соединение восстановлено и есть средства.

{

machine.Alert?.Invoke("Банкомат разблокирован. Переход в режим ожидания.");

machine.CurrentState = new IdleState(); // Переходим в режим ожидания.

}

}

}

}

using ClassLibrary; // Подключение библиотеки с логикой работы банкомата

namespace WinFormsApp

{

public partial class Form1 : Form // Основной класс формы, унаследован от базового класса Form

{

private AtmMachine atmMachine; // Поле для хранения объекта банкомата

public Form1()

{

InitializeComponent(); // Инициализация компонентов формы

atmMachine = new AtmMachine(1, 10000, true, 0.05); // Создание объекта банкомата с уникальным ID, начальными деньгами, статусом онлайн и шансом сбоя

atmMachine.Alert = message => lblStatus.Text = message; // Установка обработчика событий Alert для отображения сообщений в метке lblStatus

}

private void btnEnterPin\_Click(object sender, EventArgs e)

{

string pin = txtPin.Text; // Получение введенного пользователем PIN-кода из текстового поля txtPin

atmMachine.EnterPin(pin); // Передача PIN-кода банкомату для обработки

}

private void btnWithdraw\_Click(object sender, EventArgs e)

{

if (int.TryParse(txtAmount.Text, out int amount)) // Проверка корректности введенной суммы

{

atmMachine.ExecuteTransaction("withdraw", amount); // Попытка снять указанную сумму через банкомат

}

else

{

MessageBox.Show("Введите корректную сумму."); // Сообщение об ошибке при некорректном вводе

}

}

private void btnRefill\_Click(object sender, EventArgs e)

{

if (int.TryParse(txtAmount.Text, out int amount)) // Проверка корректности введенной суммы

{

atmMachine.RefillCash(amount); // Пополнение банкомата указанной суммой

}

else

{

MessageBox.Show("Введите корректную сумму."); // Сообщение об ошибке при некорректном вводе

}

}

private void btnExit\_Click(object sender, EventArgs e)

{

atmMachine.ExecuteTransaction("exit"); // Завершение работы и переход банкомата в режим ожидания

}

private void MainForm\_Load(object sender, EventArgs e)

{

lblStatus.Text = "Банкомат в режиме ожидания."; // Установка начального статуса банкомата при загрузке формы

}

}

}

https://github.com/Deligan/tkpo