**Міністерство освіти і науки України**
**Національний технічний університет України “Київський політехнічний інститут імені Ігоря Сікорського”**
**Факультет інформатики та обчислювальної техніки Кафедра інформаційних систем та технологій**

# Лабораторна робота №3
# Діаграма розгортання. Діаграма компонентів. Діаграма взаємодій та послідовностей
# Варіант: 25. Installer generator

Виконав:

Студент групи ІА-22

Птачик Р.С.

Перевірив:

Мягкий М.Ю.

**Київ 2024**

# Зміст

**Тема:** Діаграма розгортання. Діаграма компонентів. Діаграма взаємодії та послідовностей.

**Мета:** Ознайомитися з теоретичними відомостями. Створити діаграму розгортання, діаграму компонентів, діаграму взаємодії та послідовностей. Реалізувати частину функціональності системи, описану діаграмами з попередньої лабораторної роботи згідно обраної теми.

## Хід роботи:

## Тема:

### 25. Installer generator (iterator, builder, factory method, bridge, interpreter, client-server)

Генератор інсталяційних пакетів повинен мати якийсь спосіб налаштування файлів, що входять в установку, установки вікон з інтерактивними можливостями (галочка - створити ярлик на робочому столі; ввести в текстове поле деякі дані, наприклад, ліцензійний ключ і т.д.). Генератор повинен вивести один файл .exe або .msi.

## Короткі теоретичні відомості

### Діаграма розгортання

Діаграма розгортання відображає обчислювальні вузли, компоненти та об'єкти, які функціонують під час роботи програми. Вона представляє робочі екземпляри компонентів – одиниць коду, що виконуються на цих вузлах. Компоненти, які не мають робочих екземплярів під час виконання, не показуються на діаграмі розгортання, а можуть бути відображені на діаграмі компонентів, де фокус зроблено на зв’язках між типами компонентів.

У UML діаграма розгортання моделює фізичне розміщення артефактів на обчислювальних вузлах. Вузли зображаються у вигляді прямокутних паралелепіпедів із вкладеними в них артефактами, які представлені прямокутниками. Вузли можуть містити підвузли, відображені як вкладені паралелепіпеди. Один вузол може концептуально відповідати множині фізичних вузлів, наприклад, кластеру серверів баз даних.

Існує два типи вузлів:

- **Вузол пристрою** — фізичний обчислювальний ресурс із власною пам'яттю та сервісами для виконання програмного забезпечення, як-от ПК, мобільні телефони тощо.
- **Вузол середовища виконання** — програмний обчислювальний ресурс, який працює всередині іншого вузла та надає сервіси для виконання інших програмних елементів.

## Діаграма компонентів

Діаграма компонентів UML показує систему, розділену на окремі модулі. Існують три основні типи таких діаграм:

1. **Логічні**
2. **Фізичні**
3. **Виконувані**

Найчастіше використовують логічне розбиття, де система представлена як набір незалежних компонентів, що взаємодіють між собою. Наприклад, система продажу товарів може складатися з таких компонентів, як каса, сервер продажів, система обліку тощо. Ці компоненти можуть розміщуватися на різних фізичних пристроях або в окремих процесах.

**Фізичне розбиття** акцентує увагу на розподілі компонентів між серверами та комп'ютерами, і найчастіше зображується на діаграмі розгортання.

**Виконуване розбиття** показує компоненти як файли (наприклад, .exe, html, бази даних), пропонуючи інший погляд на систему. Однак такий підхід використовується рідше через наявність діаграм розгортання.

Компоненти повинні забезпечувати гнучкість для користувачів, дозволяючи оновлювати систему частинами та підтримувати сумісність з рішеннями різних виробників, хоча це іноді складно реалізувати.

## Діаграма послідовностей

Діаграма послідовностей — це графічне відображення взаємодії об'єктів у часі. Вона показує задіяні об'єкти та послідовність повідомлень, які вони надсилають один одному. Така діаграма ілюструє час передачі та отримання повідомлень, демонструючи динаміку сценаріїв використання в системі.

Ці діаграми корисні для деталізації діаграм прецедентів і опису логіки сценаріїв. Вони є відмінним засобом для документування проекту з точки зору користувацьких сценаріїв, оскільки відображають об'єкти, що взаємодіють, повідомлення, які передаються, та результати, що повертаються у відповідь на повідомлення.

На діаграмі послідовностей зображують лише об'єкти, які беруть участь у певній взаємодії. Лінія життя об'єкта позначається вертикальною пунктирною лінією, що асоційована з об'єктом і вказує на тривалість його існування у системі. Якщо об'єкт є постійним у системі, його лінія життя тягнеться від верхньої до нижньої межі діаграми.

Під час функціонування об'єктно-орієнтованої системи об'єкти можуть бути активними (виконують певні дії) або пасивними (очікують повідомлень). Для відображення активності використовується поняття "фокус управління" в UML. Це витягнутий вузький прямокутник, який вказує період активності об'єкта. Верхня межа прямокутника означає початок фокусу, а нижня — його завершення. Він розташовується під об'єктом та може замінювати його лінію життя, якщо об'єкт активний протягом всього часу.

Взаємодія між об'єктами описується повідомленнями, які вони обмінюються між собою. Повідомлення — це завершений фрагмент інформації, надісланий одним об'єктом іншому, яке ініціює виконання певних дій для розв'язання конкретного завдання.

## Реалізація попередньої лабораторної роботи

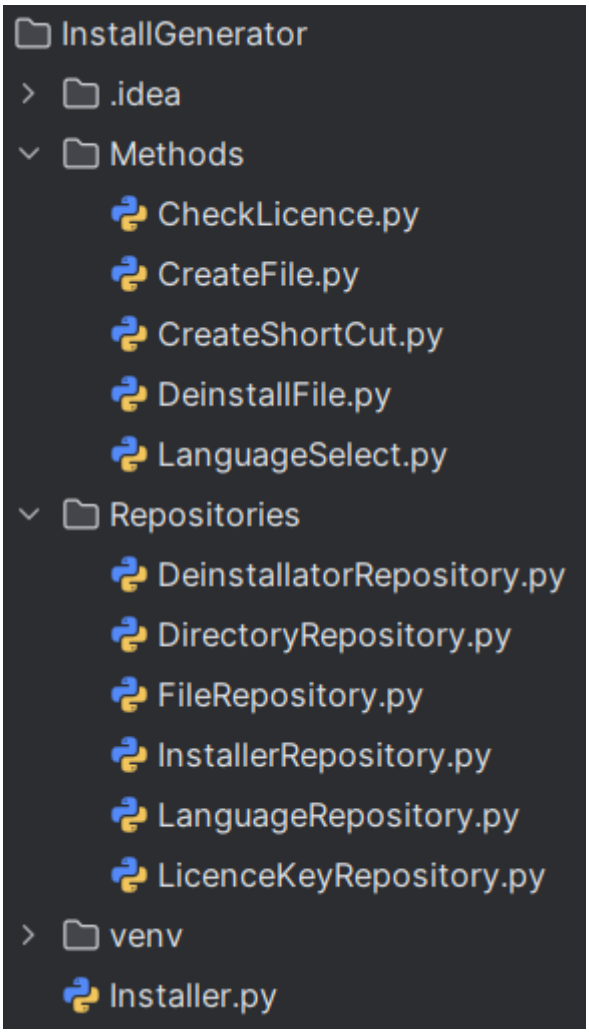


Рис. 1 – Реалізація попередньої ЛР

На рис. 1 зображено реалізацію частини функціональності системи, описану діаграмами з попередньої лабораторної роботи згідно обраної теми.

**Структура проекту:**

**Пакет Repositories (містить файли для збереження даних):**

- **DirectoryRepository:** Містить дані про шлях до файлів
- **DeinstallatorRepository:** Зберігає дані про інсталятор – ім’я деінсталятора та шлях до деінсталятора
- **FileRepository:** Зберігає дані про встановлювальний файл – ім’я файлу та шлях до файлу
- **LanguageRepository:** Містить мови для інсталятора
- **LicenceKeyRepository:** Містить ліцензійний ключ
- **InstallerRepository:** Зберігає в собі всі дані з попередніх репозиторіїв

**Пакет Methods (містить основні методи інсталятора):**

- **CheckLicence:** Перевіряє чи введений користувачем ліцензійний ключ вірний
- **CreateFile:** Створює файл з розширенням (.exe або .msi) та створює деінсталятор
- **CreateShortCut:** Створює ярлик на робочому столі
- **DeinstallFile:** Метод для видалення файлу та деінсталятора
- **LanguageSelect:** Метод який вказує мову інтерфейсу інсталятора

**Файл Installer:** файл в якому дані передаються в методи і в якому ці методи використовуються.

## Діаграма взаємодії та послідовностей

Діаграма послідовностей **для збереження файлу**:

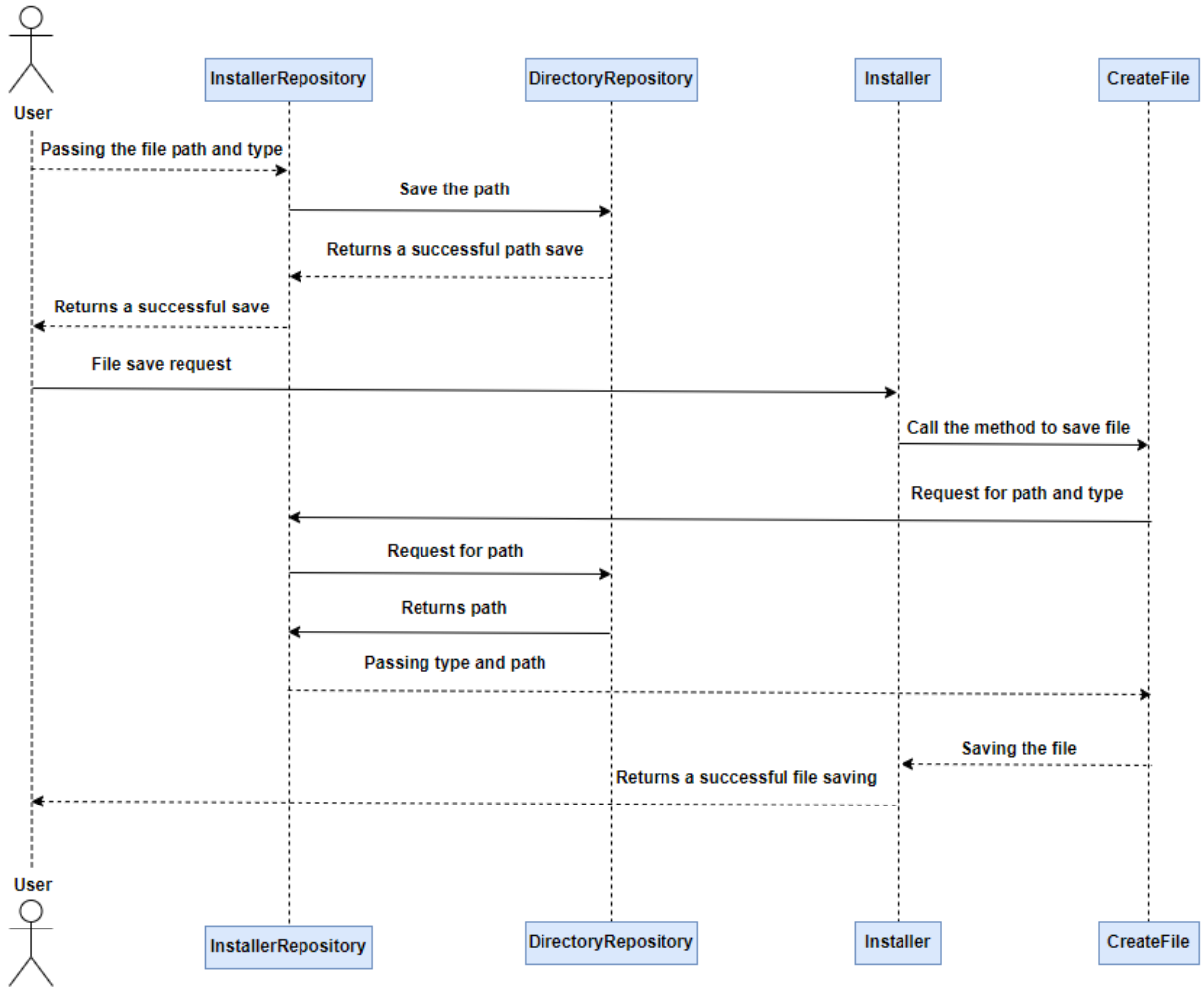


Рис. 2 – Діаграма взаємодії для збереження файлу

**Збереження даних про шлях до директорії та тип файлу:**

1. Користувач вибирає в яку директорію встановити файл та вибирає тип файлу.

- Якщо користувач не вибирає директорію та тип файлу, то файл буде збережено в базовій директорії та з базовим типом файлу

2. Дані передаються в InstallerRepository:

- Тип файлу зберігається в InstallerRepository.
- Шлях до директорії зберігається в DirectoryRepository.

**Збереження файлу на пристрої:**

1. Користувач надає запит на збереження файлу на пристрої.

2. Installer викликає метод CreateFile для збереження файлу.

3. CreateFile надає запит на дані з InstallerRepository – тип файлу та шлях директорії.

4. InstallerRepository надає запит в DirectoryRepository на передачу даних про шлях.

5. Дані з DirectoryRepository передаються в InstallerRepository.

6. InstallerRepository передає дані про тип файлу та шлях до директорії методу CreateFile.

7. CreateFile зберігає файлу на пристрої користувача та інформує Installer про успішне збереження файлу.

8. Installer інформує користувача про успішне збереження файлу.

## Діаграма розгортання

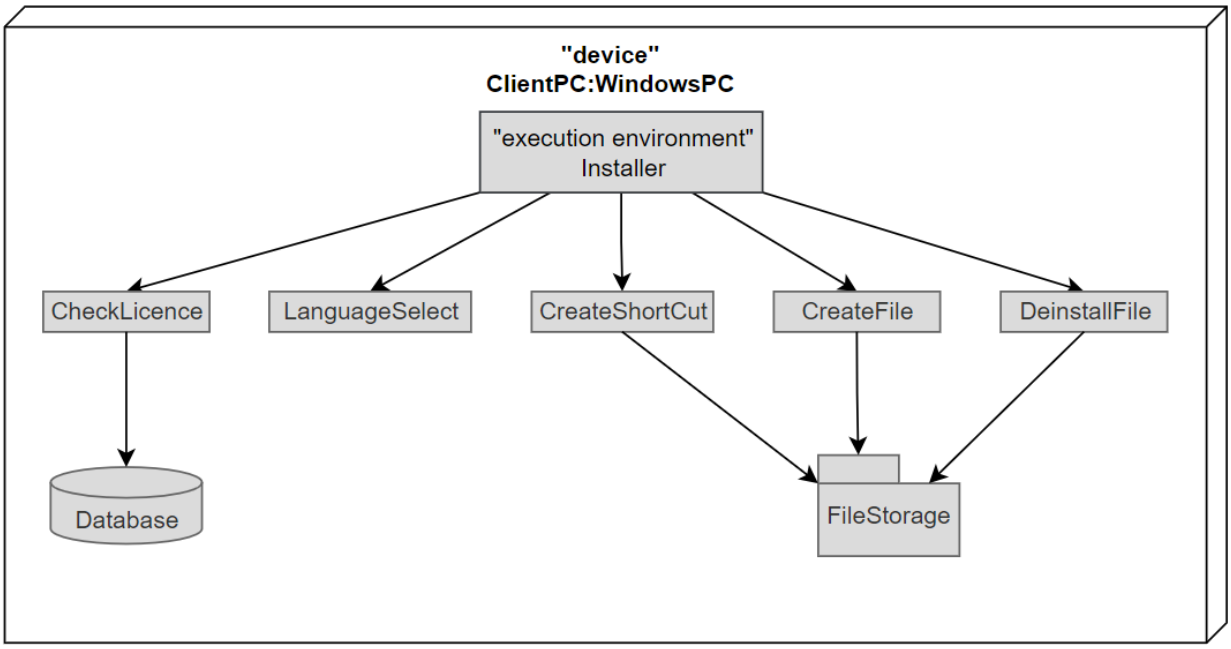


Рис. 3 – Діаграма розгортання

Елементи діаграми розгортання:

**1. Вузли:**

- **ClientPC:** фізичний комп'ютер користувача з ОС Windows, на якому запускається інсталятор, який встановлює файл на цей самий ПК.

**2. Артефакти:**

- **Installer:** головний виконавчий файл інсталятора.
- **CheckLicence, LanguageSelect, CreateShortCut, CreateFile, DeinstallFile:** методи інсталятора, які отримують дані від користувача та виконують певні операції.
- **База даних:** зберігання даних про інсталяцію, ліцензії та конфігурацій.
- **Файлова система:** потрібна для збереження/створення файлів.

**3. Зв'язки між компонентами:**

- Взаємодія між користувачем та методами через виконавчий файл інсталятора.
- Ліцензійний ключ зберігається в базі даних, а метод CheckLicence перевіряє чи вірний введений користувачем ключ.

- Встановлення файлу, деінсталятора та, якщо користувач обрав, створення ярлика, тобто взаємодія з файловою системою, відбувається завдяки методам CreateShortCut, CreateFile, DeinstallFile

## Діаграма компонентів

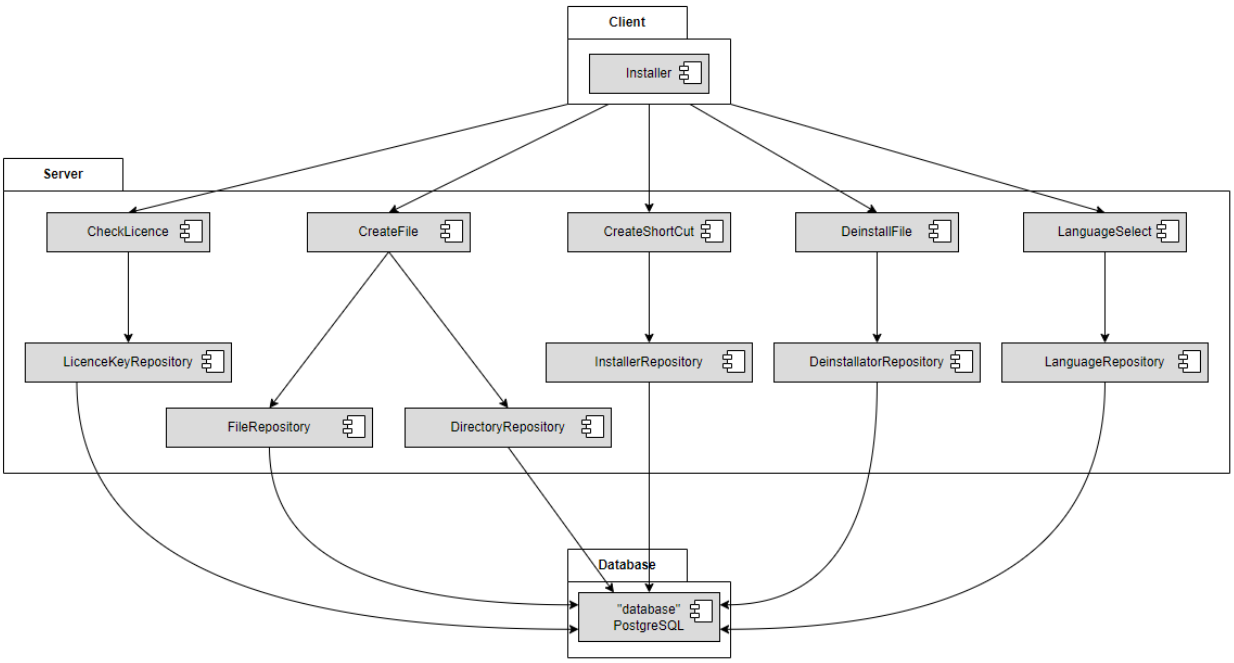


Рис. 4 – Діаграма компонентів

Діаграма компонентів демонструє взаємодію компонентів в межах самого інсталятора. Користувач взаємодіє з інсталятором, інсталятор взаємодіє з методами для встановлення файлу, а методи зберігають/отримують дані з бази даних через репозиторії.

**1. Client:**

- **Installer:** відповідає за взаємодію між користувачем та інсталятором. Installer напряму звертається до методів на стороні сервера.

**2. Server:**

- **CheckLicence, LicenceKeyRepository:** перевірка чи введений користувачем ліцензійний ключ правильний. LicenceKeyRepository бере з бази даних ліцензійний ключ та передає CheckLicence для перевірки.
- **CreateFile, FileRepository, DirectoryRepository:** встановлення файлу на пристрій користувача. FileRepository надає файл та назву файлу для встановлення, DirectoryRepository надає шлях куди потрібно

встановити файл, а CreateFile власне встановлює файл на пристрій користувача.

- **CreateShortCut, InstallerRepository:** створення ярлика на робочому столі. InstallerRepository надає інформацію чи потрібно створювати ярлик. Якщо ж користувач вибрав цю опцію, то CreateShortCut створює ярлик на робочому столі.
- **DeinstallFile, DeinstallationRepository:** видаляє файли. DeinstallationRepository містить назву деінсталятора та папку, в якій потрібно видалити файли. DeinstallFile видаляє файли.
- **LanguageSelect, LanguageRepository:** вибір мови інтерфейсу інсталятора. LanguageRepository містить мови, які можна використати, а LanguageSelect змінює мову інтерфейсу, отримуючи дані з LanguageRepository.

**3. Database:**

- **PostgreSQL:** зберігає дані в локальній базі даних.

## Код та висновок

Код можна знайти за посиланням:
https://github.com/FryMondo/TRPZ/tree/master/InstallGenerator

**Висновок:** на цій лабораторній роботі я реалізував частину функціональної системи, описану в попередній ЛР, створив діаграму розгортання, діаграму компонентів, діаграму взаємодії та послідовностей.