



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
Технологія розроблення програмного забезпечення
«Shell (total commander)»
Варіант 18

Виконав
студент групи ІА-13
Окаянченко Давид Олександрович

Перевірив:
Мягкий Михайло
Юрійович

Київ 2023р.

Мета: Навчитися розробляти діаграму розгортання. діаграма компонентів. діаграма взаємодій та послідовностей.

Завдання:

1. Ознайомитися з короткими теоретичними відомостями.
2. Розробити діаграму розгортання для проекрованої системи.
3. Розробити діаграму компонентів для проекрованої системи.
4. Розробити діаграму послідовностей для проекрованої системи.
5. Скласти звіт про виконану роботу

Варіант:

18. Shell (total commander) (state, prototype, factory method, template method, interpreter, client-server)

Оболонка повинна вміти виконувати основні дії в системі - перегляд файлів папок в файлової системі, перемикання між дисками, копіювання, видалення, переміщення об'єктів, пошук.

Хід роботи

Діаграма розгортання

Діаграма розгортання (Deployment Diagram) - це один з типів діаграм UML (Unified Modeling Language), який використовується для моделювання фізичної структури та розміщення компонентів програмної системи на апаратному обладнанні, такому як сервери, комп'ютери, сенсори, мережеві пристрої тощо. Діаграма розгортання допомагає візуалізувати, як компоненти програмної системи розташовані на різних об'єктах обладнання та як вони взаємодіють між собою через мережу.

Вузли (Nodes): Вузли представляють фізичні або віртуальні об'єкти обладнання, такі як сервери, робочі станції, маршрутизатори, бази даних, сенсори тощо. Кожен вузол може мати атрибути, такі як IP-адреси або характеристики обладнання.

Артефакти (Artifacts): Артефакти представляють програмні компоненти або файли, які розгортаються на вузлах. Це можуть бути виконувані файли, конфігураційні файли, бази даних, додатки тощо.

Зв'язки (Connections): Зв'язки показують, як компоненти взаємодіють між собою на різних вузлах. Зв'язки можуть позначати мережеві з'єднання, передачу даних, доступ до служб тощо.

Діаграми розгортання корисні для інженерів програмного забезпечення та системних архітекторів для візуалізації архітектурної концепції системи, розміщення компонентів на конкретних серверах чи обладнанні.

Компоненти: Це основні модулі або блоки, які складають систему або програмний продукт. Кожен компонент зазвичай виконує певну функцію і може бути реалізований як окремий програмний модуль або об'єкт.

З діаграми можна зрозуміти, що користувач буде взаємодіяти з додатком, що знаходиться на комп'ютері клієнта, за допомогою клавіатури, миші та монітора. Shell (наш додаток) взаємодіє з локальною файловою системою, яка дозволяє виконувати дії над файлами. Додаток також обмінюється інформацією з веб-сервером у випадку, коли користувач хоче зберегти файли в інтернеті. Веб-сервер за допомогою інтерфейсу бази даних (має зв'язок з базою даних MySQL на окремому сервері для зберігання файлів).

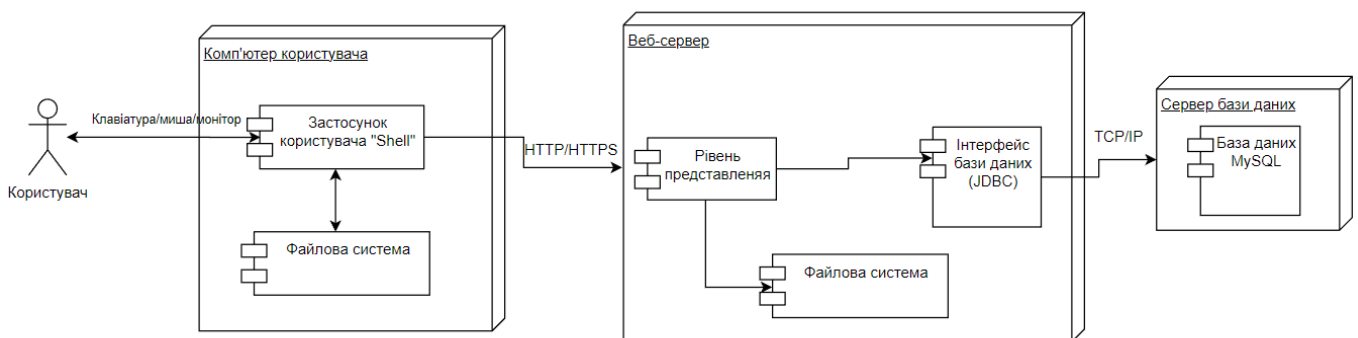


Рис. 1 Діаграма розгортання

Діаграма компонентів

Діаграма компонентів (іноді також називається діаграмою компонентної структури) - це тип діаграми, яка використовується в інженерії програмного забезпечення для візуального представлення архітектури системи або програмного продукту та її компонентів (або модулів). Діаграми компонентів допомагають розуміти, як система складається з окремих частин

(компонентів), як вони взаємодіють між собою та як вони спільно працюють для досягнення цілей системи.

Компоненти: Це основні модулі або блоки, які складають систему або програмний продукт. Кожен компонент зазвичай виконує певну функцію і може бути реалізований як окремий програмний модуль або об'єкт.

Зв'язки: Діаграми компонентів включають зв'язки між компонентами, що показують, як вони взаємодіють між собою. Зв'язки можуть представляти залежності між компонентами, наприклад, використання одного компонента іншим, або вони можуть вказувати на комунікацію між компонентами через інтерфейси.

Інтерфейси: Компоненти можуть мати інтерфейси, які описують спосіб взаємодії з іншими компонентами. Інтерфейси вказують, які методи або функції можуть бути викликані із зовнішніх компонентів.

Залежності: Діаграми компонентів можуть також показувати залежності між компонентами, що вказують на те, як один компонент може впливати на інший.

У моєму випадку ми можемо спостерігати такі компоненти як: комп'ютер користувача, який використовує Shell (наш додаток). Shell використовує файлову систему комп'ютера для виконання дій з файлами. Shell має зв'язок з веб-сервером, до якого звертається додаток. Існує також база даних, яка надає послуги веб-серверу для зберігання файлів.

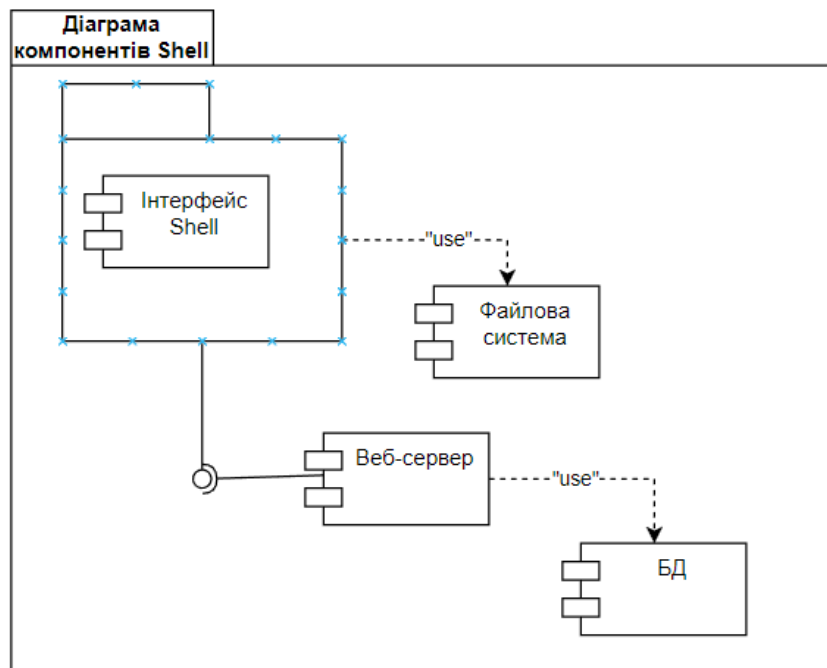


Рис. 2 Діаграма компонентів

Діаграми послідовностей

Ці діаграми використовуються для візуалізації взаємодії між об'єктами або компонентами в системі впродовж певного проміжку часу. Вони дозволяють показати, як об'єкти взаємодіють між собою, обмінюючи повідомленнями або виконуючи методи, і як ця взаємодія впливає на стан системи.

Основні елементи діаграм послідовностей включають:

Об'єкти (Objects): Об'єкти або сутності, які беруть участь у взаємодії, представлені у верхній частині діаграми. Кожен об'єкт позначається ім'ям і може мати життєвий цикл, який відображається на діаграмі зліва направо.

Лінії життєвого циклу (Lifelines): Лінії життєвого циклу об'єкта відображають, як триває його існування під час взаємодії. Вони позначаються вертикальними лініями, на яких розташовані повідомлення, інтервали активації та інші моменти життєвого циклу об'єкта.

Повідомлення (Messages): Повідомлення показують передачу інформації або виконання дій між об'єктами. Існують різні типи повідомлень, такі як синхронні (з блокуванням), асинхронні (без блокування), відповіді на повідомлення тощо.

Діаграми послідовностей корисні для аналізу та проектування взаємодії в системі. Вони допомагають визначити порядок виконання операцій та повідомлень між об'єктами та можуть бути використані для детального опису архітектурних взаємодій в програмному забезпеченні перед його реалізацією.

У даному випадку діаграму послідовностей було реалізовано на прикладі операції переіменування файлу. Ми можемо спостерігати, що спочатку користувач має обрати файл для перейменування. Інтерфейс програми має відобразити випадаючий список з можливими діями. Користувач натискає «перемейнувати» і вводить нове ім'я. Після натискання кнопки «Enter», запит передається на Shell, який, в свою чергу, звертається до файлової системи. Файлова система змінює назву файлу і повертає результат, який через інтерфейс бачить користувач.

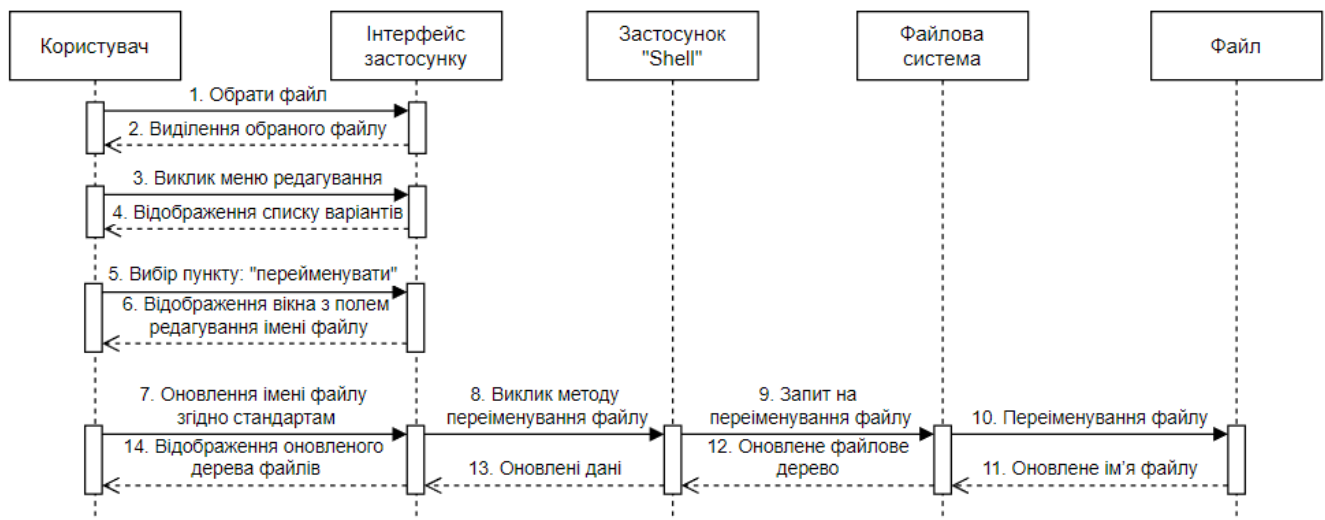


Рис. 3 Діаграма послідовностей сценарію «переіменування файлу»

Висновок: У ході виконання лабораторної роботи було проведено ознайомлення з теоретичними відомостями та розроблено діаграма розгортання, діаграма компонентів та діаграма послідовностей. Окрім того, підготовлений звіт включає всі необхідні компоненти, що відображають структуру розробленої системи.