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

ЛАБОРАТОРНА РОБОТА №8

З дисципліни «Технологія розробки програмного забезпечення»

Тема: «ШАБЛОНИ «COMPOSITE», «FLYWEIGHT», «INTERPRETER», «VISITOR»»

Виконав:
студент 3 курсу,
групи ІА-12
Одемчук Н. О.

Перевірив:
ас. Колеснік В.М.

Київ 2023

Тема: ШАБЛОНИ «COMPOSITE», «FLYWEIGHT», «INTERPRETER», «VISITOR»
Мета: Застосування одного з розглянутих шаблонів при реалізації програми.

Завдання:

Клієнт для IRC-чатів з можливістю вказівки порту і адреси з'єднання, підтримка базових команд (підключення до чату, створення чату, установка імені, реєстрація, допомога і т.д.), отримання метаданих про канал.

Хід роботи

Паттерн Композит використовується у наданому коді для створення структури, де окремі команди та комбіновані команди (що складаються з декількох команд) обробляються однаково. Цей паттерн дозволяє будувати складні структури, використовуючи прості окремі об'єкти та об'єднувати їх у великі структури.

Ось як застосовується паттерн Композит у наданому коді:

1. Абстрактний базовий клас (`Command`):
    - `Command` - це абстрактний базовий клас (ABC), який визначає загальний інтерфейс для всіх конкретних класів команд.
    - Він оголошує абстрактний метод `execute()`, який повинен бути реалізований конкретними підкласами.

2. Листові вузли (`JoinChannelCommand`, `LeaveChannelCommand`, і т.д.):
    - Кожна конкретна команда, така як `JoinChannelCommand`, `LeaveChannelCommand` і т.д., розширює базовий клас `Command` та реалізує метод `execute`.
    - Ці класи представляють листові вузли композитної структури.

3. Композитний вузол (`CompositeCommand`):
    - `CompositeCommand` - це конкретний клас, який розширює базовий клас `Command`.
    - У нього є внутрішній список (`self.commands`), щоб зберігати колекцію об'єктів `Command`, які можуть бути листовими вузлами або іншими композитними вузлами.

- Він реалізує метод `execute`, який ітерується по своєму списку команд та виконує кожну, повертаючи конкатенований результат.

4. Використання у `MessageHandler`:
    - `MessageHandler` використовує паттерн Композит для обробки як окремих команд, так і комбінованих команд.
    - Клас `CompositeCommand` використовується для агрегації та виконання декількох команд, коли користувач надає список команд, розділених крапкою з комою.
    - Метод `command_choice` перевіряє, чи містить введене повідомлення декілька команд (визначається наявністю ';'). У разі позитивної відповіді він створює об'єкт `CompositeCommand` та додає до нього окремі команди.

Цей паттерн дозволяє вам обробляти окремі команди та комбіновані команди однаково. Незалежно від того, чи є команда окремою операцією чи складається з кількох операцій, клієнтський код (у цьому випадку, `MessageHandler`) може виконувати їх у єдинообразний спосіб. Це сприяє гнучкості та масштабованості при обробці складних структур команд.

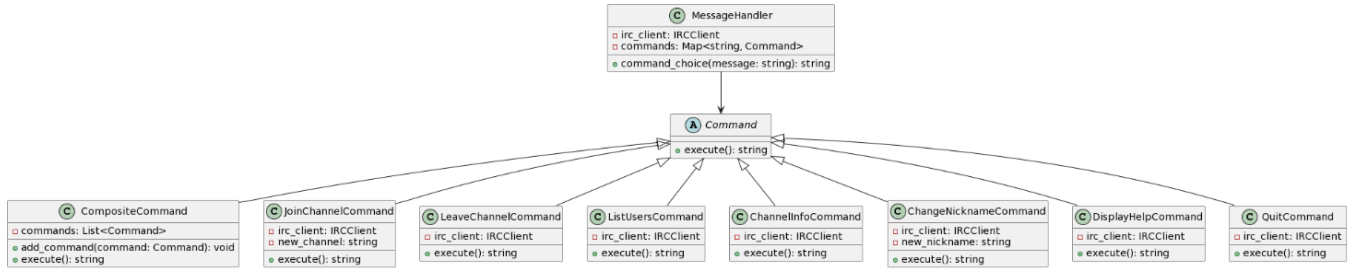

Рисунок 8.1 – Діаграма шаблону «COMPOSITE»

Висновки: У цій лабораторній роботі я реалізував шаблон «COMPOSITE».