Є. А. Васянович
Д.І. Катєльніков

ВИКОРИСТАННЯ ШАБЛОНІВ ПРОЕКТУВАННЯ У СУЧАСНОМУ ПІДХОДІ ДО РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Вінницький національний технічний університет

Анотація

В даній роботі розглянуто значимість шаблонів проектування у сучасній розробці програмного забезпечення. Також розглянуто найпопулярніші шаблони проектування, їх класифікацію, переваги, недоліки та приклади використання.

Ключові слова: шаблони проектування, сінглтон, спостерігач, фасад, GoF.

Abstract

In this work was reviewed the importance of design patterns in modern software development. The most popular design templates, their classification, advantages, disadvantages and examples of use are also considered.

Keywords: design patterns, singleton, observer, facade, GoF.

Вступ

Сучасна розробка програмного забезпечення є складним та творчим процесом. Перш ніж почати створювати будь-яку програму необхідно розробити її архітектуру, інакше ПЗ може виявитися неефективним, понесе багато витрат та може взагалі провалитися на ринку. Саме тому треба дуже ретельно підійти до питання вибору архітектури розроблюваного ПЗ.

Існує багато різних технік та рішень при розробці програмного забезпечення. Це можуть бути різні правила, принципи та певні структури. Усе більше в наш час потребують знання шаблонів проектування. Проте, щоб ефективно розробляти програмне забезпечення, потрібно знати та вміти їх застосовувати, адже не можна їх використовувати усі одразу у будь-якій програмі. Тому сучасному розробнику потрібно знати які шаблони проектування, як і де краще застосувати.

Огляд найпоширеніших шаблонів проектування

Перші ніж розглядати шаблони проектування варто розібратися у їх класифікації. В загальному їх поділяють на 3 групи: породжуючі, структурні та поведінкові. Також їх поділяють за рівнем на ті, що працюють на рівні класу та ті, які працюють з об’єктом. Поділ основних шаблонів проектування GoF(Gang of Four) можна побачити на рисунку 1 [1].

Мета / Рівень	Породжуючі патерни	Структурні патерни	Патерни поведінки
Клас	Factory Method	Adapter (класу)	Interpreter Template Method
Об'єкт	Abstract Factory Singleton Prototype Builder	Adapter (обєкту) Decorator Proxy Composite Bridge Flyweight Facade	Iterator Command Observer Visitor Mediator State Strategy Memento Chain of Responsibility

Рис. 1 – Класифікація шаблонів проектування Gof

Розглянемо три з найпопулярніших шаблони проектування (по 1 з кожної групи), що можуть дуже часто зустрічатись та використовуватись при розробці ПЗ.

Першим шаблоном, що буде розглянуто – це Singleton. Даний шаблон є дуже відомим і простим в реалізації, хоча в той же час і підступним. Його основна ціль – надати єдину точку доступу до класу при цьому гарантуючи одночасне існування лише одного екземпляру даного класу. Він є дуже корисним, коли потрібно мати доступ до якогось спільного ресурсу. Проте дуже важливо його правильно реалізувати, адже даний патерн має задовольняти дві головні умови. По-перше він справді має мати лише один екземпляр (потрібно наприклад врахувати багатопоточність програми, яка може створити декілька екземплярів). По-друге потрібно реалізувати правильний доступ до об'єкту (через певний метод доступу, а не через глобальну змінну). Тільки при дотриманні цих вимог можна отримати користь від даного шаблону. До переваг можна віднести зручність розробки програми при його використанні та відносну простоту реалізації. Недоліками є порушення першого принципу SOLID (принцип єдиного обов'язку). На рисунку 2 зображено діаграму даного шаблону проектування [2].

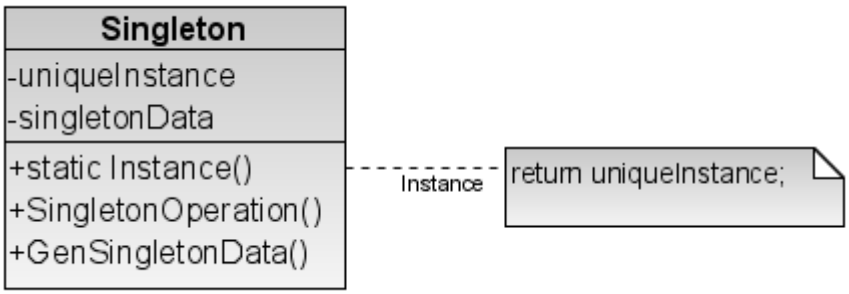

Рис. 2 – Діаграма класів структури дизайн-патерну "Singleton"

Наступним шаблоном, що розглядається буде патерн Observer ("Спостерігач"). Даний шаблон є досить поширеним серед розробників, а складність його реалізації не висока. Мета даного шаблону – створення у програмі системи взаємодії між об'єктами за допомогою підписок та сповіщень про подій. Observer допомагає реалізувати відношення між класами "один до багатьох", при цьому реалізовуючи слабкий зв'язок між класами. Основними складовими даного шаблону є клас-видавець та класи-підписники. Клас-видавець інкапсулює логіку підписок та сповіщає підписників, що відбулась певна подія і вони повинні відреагувати. Даний механізм вирішує проблему, щоб кожен клас періодично не перевіряв чи не відбулася певна подія, а натомість отримує сповіщення. Перевагами даного шаблону є слабка залежність класів та підписування на події під час виконання програми. Також в деяких мовах програмування (таких як C#, JavaScript) вже є зручні механізми для реалізації цього дизайн-патерну ("events"). Очевидних недоліків у даного шаблону немає. Саме тому його варто використовувати для того, щоб мати гарну архітектуру програми. На рисунку 3 зображено діаграму даного шаблону проектування [3].

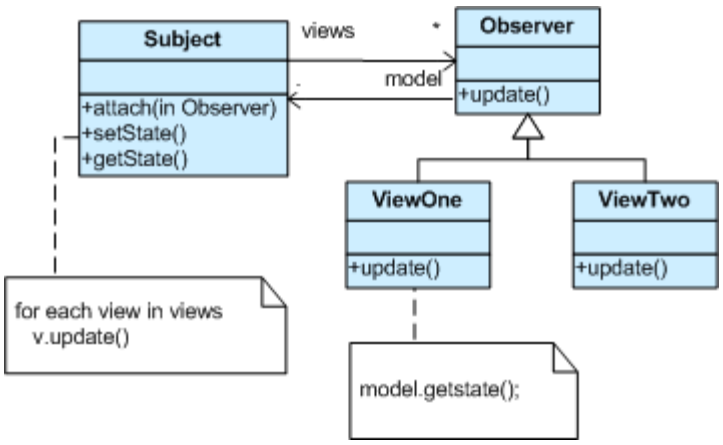

Рис. 3 – Діаграма класів структури дизайн-патерну “ Observer ”

Останнім розглянутим шаблоном проектування буде структурний патерн Facade (“Фасад”). Він є нескладним та досить поширеним у використанні. Даний шаблон дозволяє абстрагуватися від дуже складної системи класів, надаючи користувачеві простий API, з яким легко працювати та який реалізовує лише потрібний йому функціонал. Таким чином він ізолює клієнта від складної системи. В той же час можна спокійно змінювати механізми складної системи не порушуючи функціональності програми. Даний патерн також послаблює зв’язність коду. Серед недоліків можна виділити наступний: клас цього шаблону може перетворитися на “божественний клас” (тобто той, що виконує надто багато функції, порушуючи перший принцип SOLID). Таким чином потрібно з обережністю використовувати цей шаблон проектування. На рисунку 4 зображено діаграму шаблону проектування Facade[4].

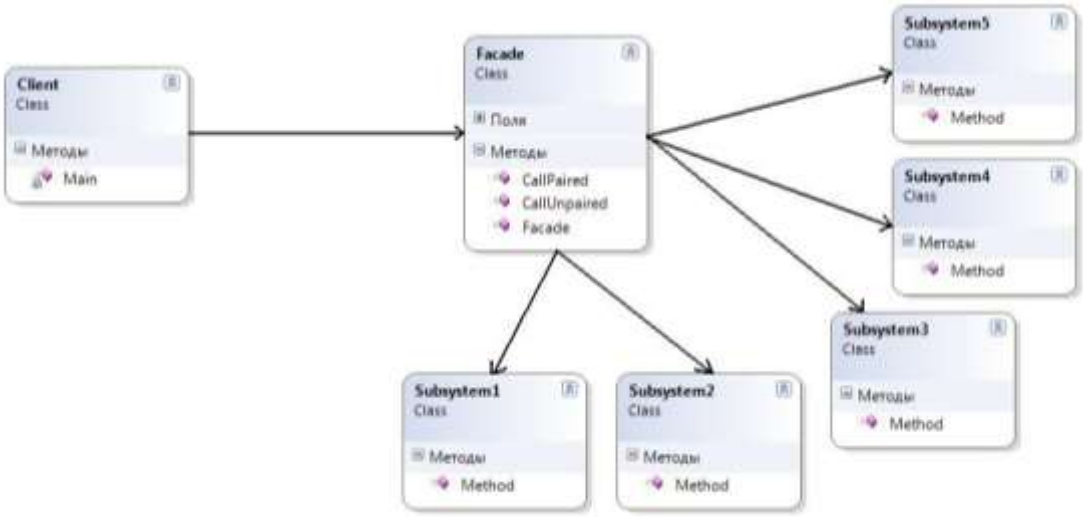

Рис. 4 – Діаграма класів структури дизайн-патерну “Facade”

Висновки

Таким чином, було продемонстровано важливість шаблонів проектування у сучасній архітектурі програмного забезпечення. Кожен шаблон має свою область використання та свої недоліки та переваги. Проте дуже важливо знати, як вони влаштовані. Адже в будь-якому сучасному додатку можна зіткнутися з певним дизайн-патерном. Саме тому було розглянути найпопулярніші з них, що можуть часто зустрічатися. Адже, крім розробки ПЗ знання шаблонів допомагають ефективніше розуміти код та полегшують розробку у команді.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Принципи проектування – [Електронний ресурс]. – Режим доступу до матеріалу: https://ppt-online.org/178744
2. Одинак (шаблон проектування) – [Електронний ресурс]. – Режим доступу до матеріалу: https://uk.wikipedia.org/wiki/Одинак_(шаблон_проєктування)
3. Паттерн Observer – [Електронний ресурс]. – Режим доступу до матеріалу: http://cpp-reference.ru/patterns/behavioral-patterns/observer/
4. Фасад (шаблон проектування) – [Електронний ресурс]. – Режим доступу до матеріалу: https://uk.wikipedia.org/wiki/Фасад_(шаблон_проєктування)

Ввасянович Євгеній Анатолійович — студент групи 1ПІ-17б, факультет інформаційних технологій та комп’ютерної інженерії, Вінницький національний технічний університет, м. Вінниця, e-mail: vasianovych.zhenia@gmail.com

Науковий керівник – ***Катєльніков Денис Іванович***, кандидат технічних наук, доцент кафедри програмного забезпечення, Вінницький національний технічний університет, м. Вінниця, e-mail: fuzzy2dik@gmail.com.

Vasianovych Yevhenii Anatoliiovych — Faculty of Information Technologies and Computer Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: vasianovych.zhenia@gmail.com

Supervisor – ***Katielnikov Denys Ivanovych***, PhD, Associate Professor of Software Engineering Department, Vinnytsia National Technical University, Vinnytsia, E-mail: fuzzy2dik@gmail.com.