Федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»
Факультет программной инженерии и компьютерной техники
Направление подготовки 09.03.04 «Программная инженерия» –
Системное и прикладное программное обеспечение

Отчёт

По лабораторной работе №2

По архитектуре программных систем

Выполнил:
студент 3 курса
Батманов Даниил Евгеньевич
Группа: P3307
Принял:
Пёрл Иван Андреевич

Отчёт принят «__»_____2024 г.
Оценка: _____

г. Санкт-Петербург, 2024

Оглавление

Задание

Из списка шаблонов проектирования GoF и GRASP выбрать 3-4 шаблона и для каждого из них придумать 2-3 сценария, для решения которых могу применены выбранные шаблоны.

Сделать предположение о возможных ограничениях, к которым можем привести использование шаблона в каждом описанном случае. Обязательно выбрать шаблоны из обоих списков.

Выбранные шаблоны

1. GoF: Стратегия (Strategy)
2. GoF: Наблюдатель (Observer)
3. GRASP: Контроллер (Controller)
4. GRASP: Перенаправление (Indirection)

1. GoF: Стратегия (Strategy)

Описание шаблона:
Стратегия позволяет определить семейство алгоритмов, инкапсулировать их и делать взаимозаменяемыми. Это помогает избегать жесткой привязки к конкретным реализациям.

Сценарии:

1. **Реализация различных методов оплаты в интернет-магазине**
    a. *Проблема:* В магазине нужно поддерживать несколько способов оплаты (карты, PayPal, криптовалюты и т. д.).
    b. *Решение:* Создать интерфейс PaymentStrategy и несколько реализаций (CardPayment, PayPalPayment и т. д.). Клиент выбирает стратегию оплаты во время оформления заказа.
    c. Ограничения:
        i. Повышенная сложность структуры кода из-за добавления новых классов.
        ii. Трудности в управлении стратегиями, если их слишком много.
2. **Настройка поведения персонажа в игре**
    a. *Проблема:* Персонажи могут атаковать, защищаться, лечиться или выполнять другие действия, и поведение должно меняться динамически.
    b. *Решение:* Создать общий интерфейс для поведения и реализовать различные стратегии (AggressiveBehavior, DefensiveBehavior, HealingBehavior).
    c. Ограничения:
        i. Дополнительная память на хранение множества объектов-стратегий.
        ii. Сложность управления связями между стратегиями и внешними условиями игры.

2. GoF: Наблюдатель (Observer)

Описание шаблона:
Наблюдатель используется для установки зависимости «один-ко-многим» между объектами, чтобы при изменении состояния одного объекта автоматически обновлялись все зависимые.

Сценарии:

1. **Обновление интерфейса пользователя при изменении данных**
    a. *Проблема:* В приложении данные хранятся в модели, а интерфейс должен автоматически обновляться при их изменении.
    b. *Решение:* Модель реализует интерфейс «наблюдаемый объект», а интерфейсные элементы выступают наблюдателями, подписанными на изменения модели.
    c. Ограничения:
        i. Высокая зависимость между наблюдателями и наблюдаемым.
        ii. Возможные проблемы производительности при большом числе наблюдателей.
2. **Оповещение клиентов о скидках в системе рассылок**

a. *Проблема:* Интернет-магазин должен уведомлять подписчиков о новых скидках и акциях.
b. *Решение:* Подписчики (наблюдатели) получают уведомления от сервиса скидок (наблюдаемый объект).
c. *Ограничения:*
    i. Потенциальные проблемы с асинхронным уведомлением при большом числе подписчиков.
    ii. Зависимость подписчиков от формата данных, предоставляемых системой.

3. GRASP: Контроллер (Controller)

Описание шаблона:
Контроллер отвечает за обработку событий, поступающих от пользователя или системы, и делегирует выполнение необходимых операций другим объектам.

Сценарии:

1. **Обработка запросов в системе интернет-банкинга**
    a. *Проблема:* Разные виды операций (переводы, оплата счетов, запросы выписок) требуют корректного управления.
    b. *Решение:* Контроллер выступает связующим звеном между пользовательским интерфейсом и бизнес-логикой, направляя запросы соответствующим модулям.
    c. *Ограничения:*
        i. Увеличение сложности контроллера, если он обрабатывает слишком много задач.
        ii. Возможность превращения в «божественный объект» при неправильной декомпозиции.
2. **Управление процессом регистрации пользователей**
    a. *Проблема:* Регистрация может включать валидацию данных, отправку писем и создание учетной записи.
    b. *Решение:* Контроллер управляет последовательностью вызовов необходимых сервисов (валидация, отправка, создание).
    c. *Ограничения:*
        i. Высокая ответственность контроллера за разные аспекты процесса.
        ii. Потенциальная проблема при внесении изменений в процесс, если логика не делегируется.

4. GRASP: Перенаправление (Indirection)

Описание шаблона:
Шаблон направлен на использование промежуточного объекта для управления взаимодействием между другими объектами. Это позволяет ослабить связь между объектами, делая систему более гибкой и масштабируемой.

Сценарии:

1. **Прокси-сервер для управления запросами**
    a. *Проблема:* Система должна обрабатывать сетевые запросы и направлять их к нужным серверам, скрывая сложность внутренней структуры.
    b. *Решение:* Внедрить объект-посредник (прокси), который перенаправляет запросы на основе правил маршрутизации.

c. *Ограничения:*
    i. Дополнительная задержка из-за введения посредника.
    ii. Усложнение конфигурации при увеличении числа серверов.

2. **Обработка платежей через сторонние сервисы**
    a. *Проблема:* Приложение работает с несколькими платежными системами (Stripe, PayPal и т. д.), и нужно обеспечить единый интерфейс для взаимодействия.
    b. *Решение:* Ввести объект перенаправления, который определяет, какой сервис использовать для конкретного платежа.
    c. *Ограничения:*
        i. Повышенная сложность от поддержки и настройки логики перенаправления.
        ii. Возможные ошибки при неправильной обработке специфики платежных систем.
3. **Логгирование действий в приложении**
    a. *Проблема:* В приложение нужно встроить централизованную систему логирования, которая отправляет логи в файл, базу данных или внешний сервис.
    b. *Решение:* Использовать объект перенаправления для маршрутизации логов в зависимости от типа события (например, ошибки отправлять в базу, а информационные сообщения — в текстовый файл).
    c. *Ограничения:*
        i. Задержки при записи логов из-за добавления дополнительного слоя.
        ii. Усложнение отладки, если логика перенаправления содержит ошибки.

Систематизирующая таблица

Шаблон	Пример использования	Ограничения
Стратегия	Методы оплаты в магазине, поведение персонажа в игре	Усложнение структуры кода, управление множеством стратегий.
Наблюдатель	Обновление UI, оповещение о скидках	Высокая зависимость, проблемы производительности при большом числе наблюдателей.
Контроллер	Регистрация пользователей, обработка банковских операций	Перегрузка логики, превращение в «божественный объект», сложность изменений.
Перенаправление	Прокси-сервер, платежи, логгирование	Дополнительная задержка, усложнение конфигурации и отладки.