ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

"ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ"

Факультет ИСП

Кафедра ПИ

Индивидуальная работа

по курсу: «Конструирование программного обеспечения»

по теме: «Использование паттернов проектирования при создании программных продуктов»

Выполнил:

ст. гр. ПИ-19а

Саевский Олег

Проверил:

Чернышова А.В.

Московченко А.В.

ДОНЕЦК – 2022



**Краткое теоретическое описание порождающего паттерна Abstract Factory (Абстрактная фабрика)**

Абстрактная фабрика – это порождающий паттерн проектирования, который позволяет создавать группы взаимосвязанных (родственных) объектов, не специфицируя их классов. То есть, мы определяем интерфейс для создания взаимосвязанных объектов, без необходимости реализации конкретных классов.

Паттерн применяется в следующих случаях:

1) Когда система не должна зависеть от способа создания и компоновки новых объектов.

2) Когда создаваемые объекты должны использоваться вместе и являются взаимосвязанными.

**Теоретическое сведение по поведенческому паттерну Visitor (Посетитель)**

**Посетитель** – это поведенческий паттерн проектирования, который позволяет добавлять в программу новые операции, не изменяя классы объектов, над которыми эти операции могут выполняться.

Паттерн применяется в следующих случаях:

1) Когда имеется много объектов разнородных классов с разными интерфейсами, и требуется выполнить ряд операций над каждым из этих объектов.

2) Когда классам необходимо добавить одинаковый набор операций без изменения этих классов.

3) Когда часто добавляются новые операции к классам, при этом общая структура классов стабильна и практически не изменяется.

Пример использования паттерна Посетитель при проектировании игры изображён на рисунке 1.

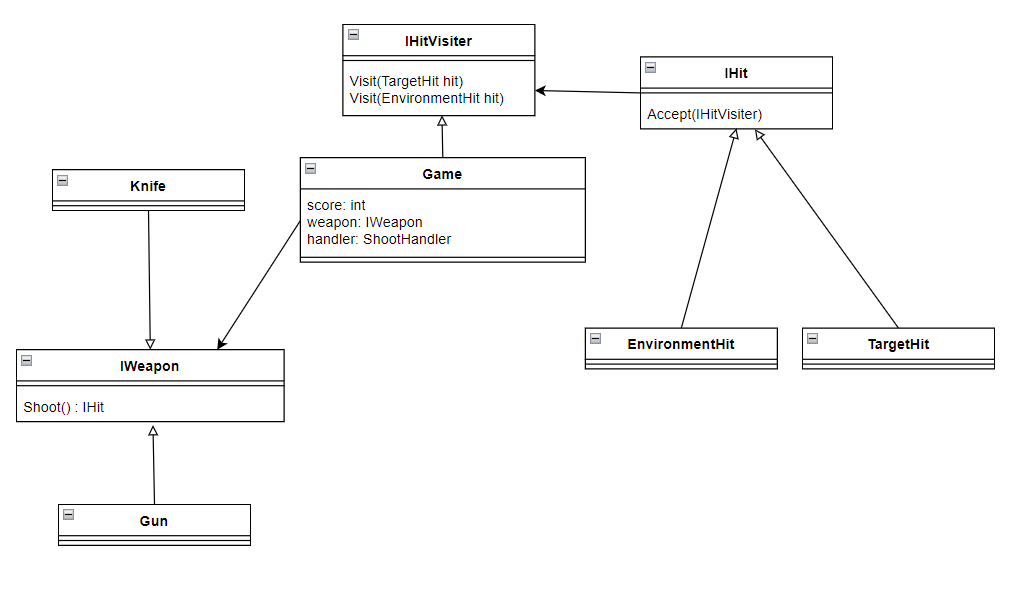


Рисунок 1 – Пример использования паттерна Посетитель

В игре существует два типа оружия: нож и ружье. При вызове метода Shoot() игрок стреляет по мишени, и возвращает объект типа IHit. Необходимо понять, увеличивать счет или нет, нужно сделать условную конструкцию if – else и проверить тип попадания (по цели или мимо). Чтобы не нагружать код данной конструкцией в случае, если типов оружий будет слишком много используется паттерн Visitor (посетитель). Класс Game реализует интерфейс IHitVisiter, что обязывает его описать поведение счета при попадании по цели и попадании мимо отдельно, и в зависимости от типа попадания будет выполнен метод с данным типом попадания. Таким образом при добавлении нового набора правил нам необходимо будет создать другой класс, реализующий интерфейс IHitVisiter и описать правила игры для каждого из типов попаданий.

**Целесообразность применения** **порождающего паттерна Abstract Factory (Абстрактная фабрика) при проектировании игры**

Например, требуется возможность генерировать различные сущности разных типов в зависимости от выбранной сложности игры. Допустим, при выборе легкого уровня игры выбирать один тип сущности, а при выборе сложного уровня – другой. В этом нам поможет паттерн Абстрактная фабрика.

**Диаграмма классов и диаграмма последовательностей без использования паттерна Abstract Factory (Абстрактная фабрика)**

Диаграммы классов и последовательностей генерации сущностей без использования паттерна Абстрактная фабрика изображены на рисунках 2-3.

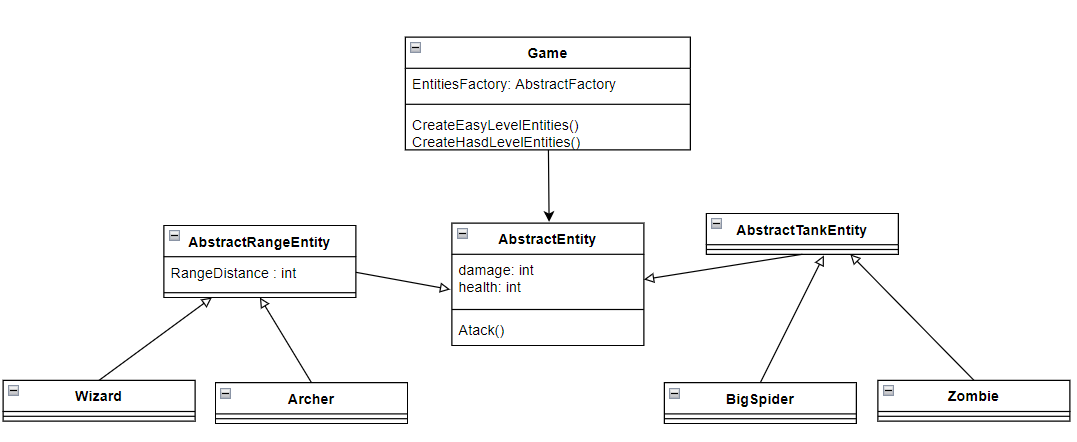


Рисунок 2 – Диаграмма классов создания объектов без использования паттерна Абстрактная фабрика

В игре есть разные сущности ближнего боя – BigSpider и Zombie, дальнего боя – Wizard, Archer. Они наследуются от классов AbstractTankEntity и AbstractRangeEntity соответственно. Класс Game занимается генерацией этих сущностей в зависимости от выбранной сложности.

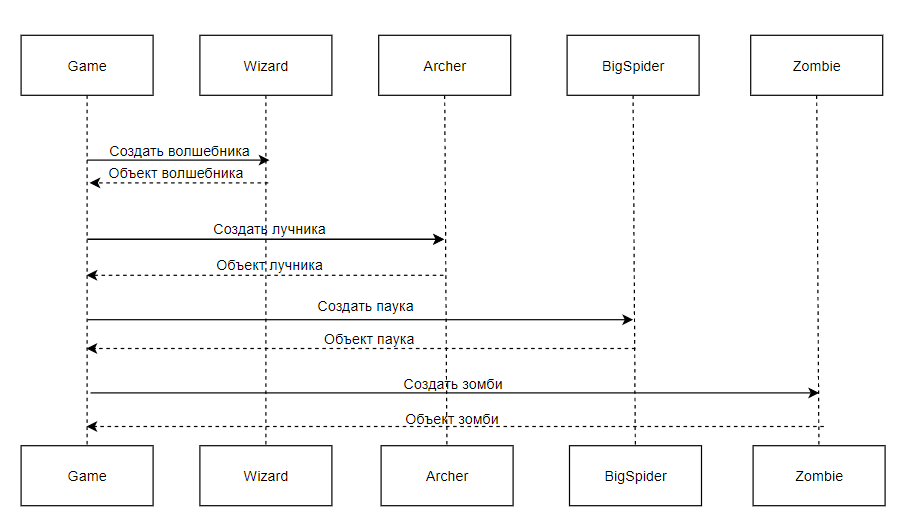


Рисунок 3 – Диаграмма последовательностей создания объектов без использования паттерна Абстрактная фабрика

**Диаграмма классов и диаграмма последовательностей с использованием паттерна Абстрактная фабрика**

Диаграммы классов и последовательностей клонирования объектов c использованием паттерна Абстрактная фабрика изображены на рисунках 4-5.

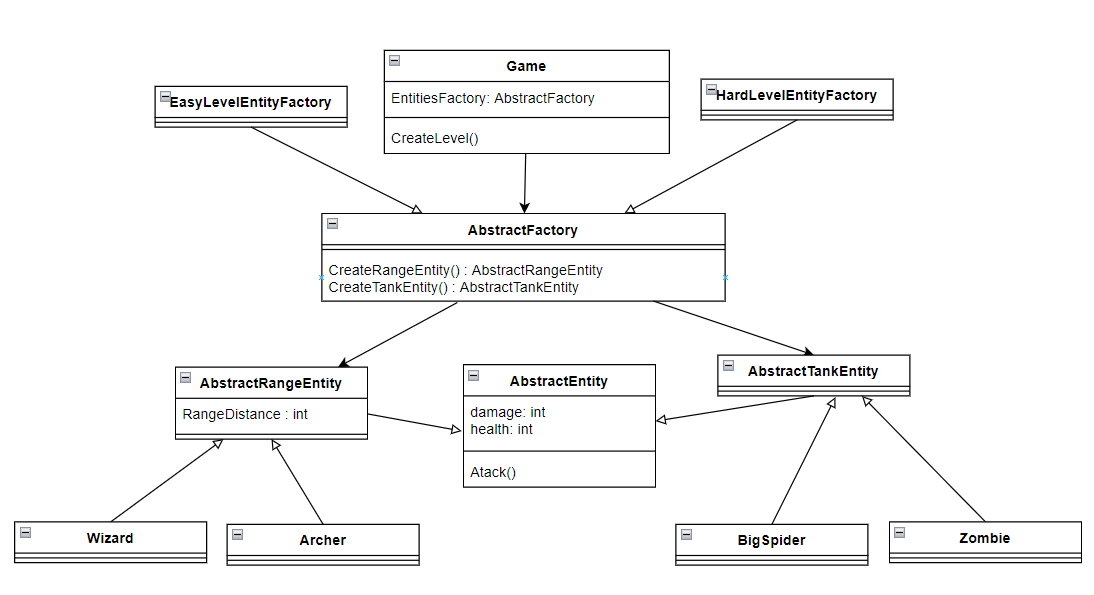


Рисунок 4 – Диаграмма классов создания объектов с использованием паттерна Абстрактная фабрика

Класс Game имеет поле с типом AbstractFactory, реализации которого занимаются генерацией объектов в зависимости от уровня сложности.

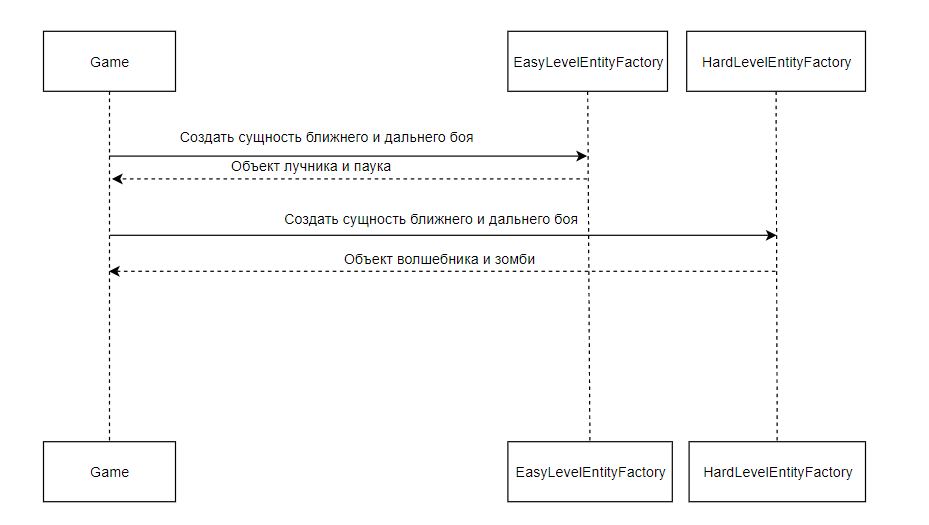


Рисунок 5 – Диаграмма последовательностей создания объектов с использованием паттерна Абстрактная фабрика

**Эффективность использования паттерна AbstractFactory (Абстрактная фабрика)**

Подход создания объектов с использованием паттерна Абстрактная фабрика имеет несколько преимуществ над подходом без его использования:

1) Методы генерации объектов для любой сложности игры строго расположены в своем классе.

2) При добавлении новых уровней сложности, достаточно просто реализовать новую фабрику и передать её в класс Game.