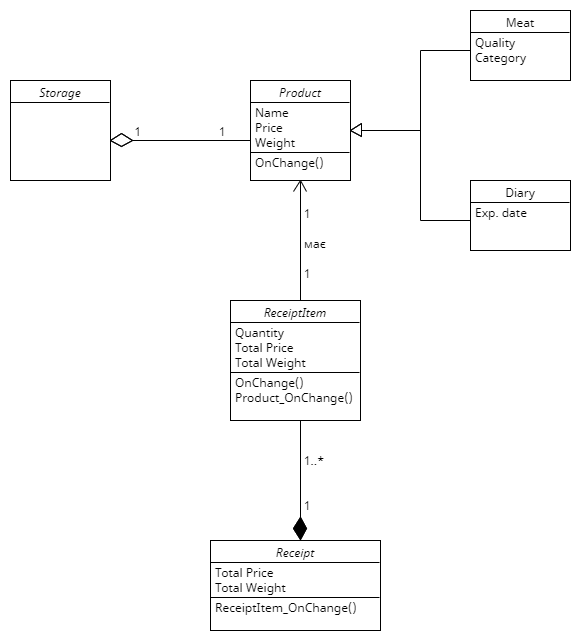
До початку проектування діаграма класів бібліотеки продуктів виглядала ось так:

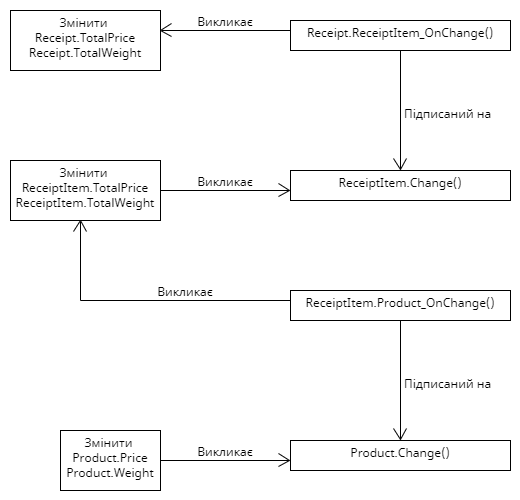


Product – абстрактний клас продукту

ReceiptItem – стрічка у чеку, замовленні із кількістю продукту

Receipt – чек зі стрічками продуктів

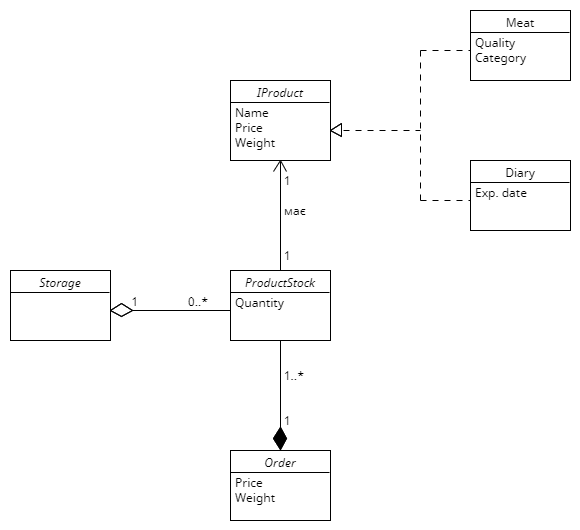
Через всю архітектуру проходить ідея ієрархії делегатів між класами Product -> ReceiptItem -> Receipt. Тобто є випадок, в якому класи вже заповненні (маємо певний чек), і якщо зміниться вага або ціна якогось елементу, то це змінить всі елементи зверху ієрархії.



Проблеми цього механізму, на мою думку:

1. Виникає потреба у абстрактному класі Product для неповторюваності коду цього механізму.
2. Потрібно піклуватися, щоб обьекти Product існували в одному екземплярі на всю програму, щоб це працювало
3. Клас Storage зберігає обьекти Product при тому, що цей механізм ніяк не відноситься до нього
4. Сумнівність взагалі потреби у такому механізмі

Нова архітектура:



IProduct – Ідея товару, який повинен мати назву, ціну та вагу

ProductStock – Сутність яка має товар та його кількість

Order – Замовлення (може також виконувати роль чеку)

Storage – Фізичний склад продуктів

Order та Storage є узагальненими (generic) через посередника ProductStock.