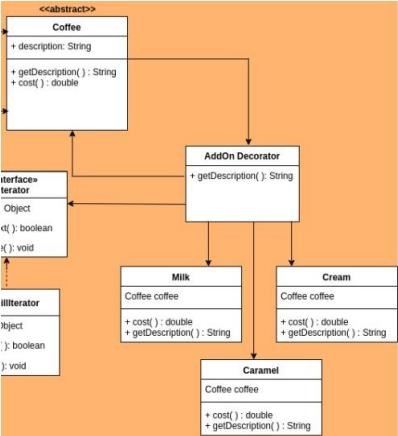
**Taller 5: Patrones**

Moisés Agudelo 202113485

**Información general del proyecto:** <https://github.com/banerjeesamrat/Virtual-Coffee-Shop>

El proyecto es una tienda virtual de café en la que los clientes tienen la posibilidad de ordenar café con base en un menú y agregar diferentes adiciones como crema batida o caramelo a su pedido por medio del patrón decorator o wrapper.

**Información y estructura del fragmento del proyecto donde aparece el patrón:**



El patron decorator se relaciona con la clase café en ambas direcciones de forma que a cada café se le puede añadir ingredientes por medio del patrón, asimismo, este se relaciona con los ingredientes que es capaz de añadir al café, que en este caso es la leche, crema y caramelo.

**Información general sobre el patrón: qué patrón es y para qué se usa usualmente:**

El patrón decorator, wrapper o decorador, es un patrón de diseño estructural cuya definición según Gamma, Helm, Johnson y Vlissides (1995) es “Añadir responsabilidades adicionales a un objeto dinámicamente. Los decoradores proveen una alternativa flexible a la subclassing para extender funcionalidades”. Este patrón resulta útil cuando se le quiere añadir responsabilidades o características adicionales a un objeto, pero no a una clase o a todos los demás objetos existentes.

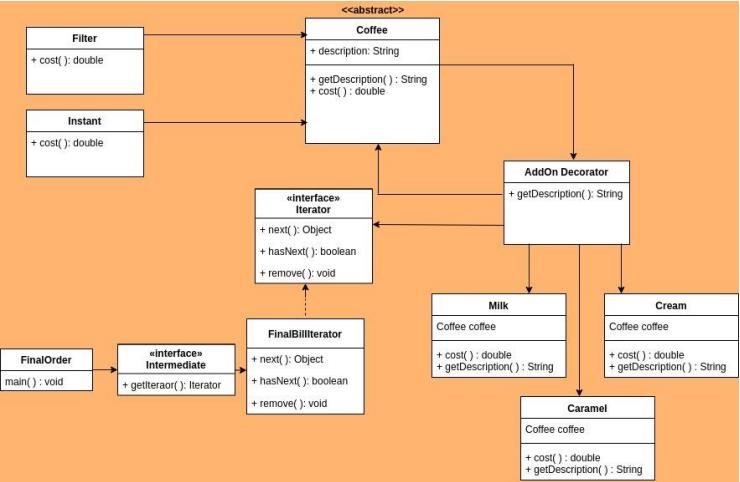
**Información del patrón aplicado al proyecto: explicar cómo se está utilizando el patrón dentro del proyecto:**

Dentro de la tienda virtual el patrón tiene como misión permitir al cliente agregar ingredientes adicionales al café que ordeno previamente. Los ingredientes que puede agregar ya están determinados en el menú y el patrón los conoce.

**¿Por qué tiene sentido haber utilizado el patrón en ese punto del proyecto? ¿Qué ventajas tiene?**

Es necesario usar el patrón decorator en este punto ya que facilita el proceso de alterar las características de un solo objeto y permite que no haya necesidad de crear más clases que representen el café con el ingrediente ya agregado, por ejemplo, gracias al patrón no hay una clase café con leche o café con caramelo o café con crema sino una sola llamada café que relega la responsabilidad de añadir ingredientes a otra parte del programa, que, en este caso, resulta ser el patrón decorator **¿Qué desventajas tiene haber utilizado el patrón en ese punto del proyecto?**

Podría decirse que el uso de este patrón aumenta la dependencia entre clases ya que de la clase decorator depende enteramente el proceso de añadir ingredientes al café, además, de acuerdo con cómo está organizado el proyecto y como se puede ver en el UML, si llegase a fallar esta clase sería imposible proseguir con el proceso de facturación.



**¿De qué otras formas se le ocurre que se podrían haber solucionado, en este caso particular, los problemas que resuelve el patrón?**

Pudo haberse agregado a la clase café un atributo de tipo ArrayList que agrupe los ingredientes que el cliente quiere que su café lleve. Se podrían agregar ingredientes por medio de un nuevo método bajo esta misma clase o sino por medio de una nueva con el fin de no asignarle responsabilidades adicionales que no están relacionadas a la clase café.