7° PRÁCTICA DE LABORATORIO

Presentado por:
Jesus Gabriel Parra Dugarte

Presentado a:
Ricardo Zambrano

Laboratorio de Ingeniería de Software II

Universidad del Cauca

Facultad de Ingeniería Electrónica y Telecomunicaciones

Programa de Ingeniería de sistemas

Popayán, Cauca

11/octubre/2023

1. Desarrollar Plantilla

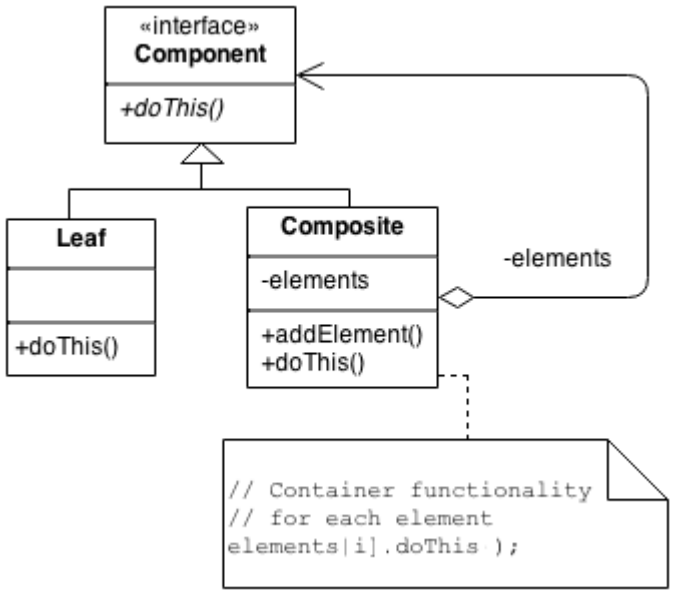
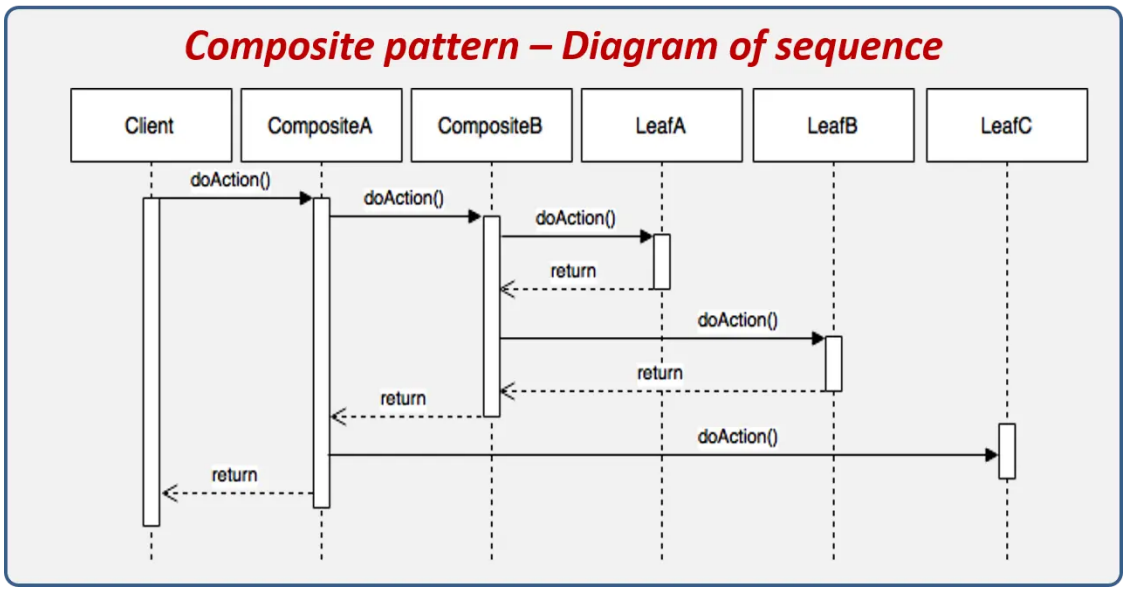
Patrón estructural: **Composite**	
Intención	permite componer objetos en estructuras de árbol y trabajar con esas estructuras como si fueran objetos individuales
Soluciona qué problema	Composite sólo tiene sentido cuando el modelo central de tu aplicación puede representarse en forma de árbol
Solución propuesta	El patrón Composite sugiere que trabajes con Productos y Cajas a través de una interfaz común. La gran ventaja de esta solución es que no tienes que preocuparte por las clases concretas de los objetos que componen el árbol. No tienes que saber si un objeto es un producto simple o una sofisticada caja. Puedes tratarlos a todos por igual a través de la interfaz común. Cuando invocas un método, los propios objetos pasan la solicitud a lo largo del árbol.
Diagrama de clases	

Diagrama de secuencias	
Participantes	Interfaz de componente, hojas y composiciones.
Aplicabilidad	Se usa para representar jerarquías reales con el uso de árboles
Consecuencias	• Puede resultar difícil proporcionar una interfaz común para clases cuya funcionalidad difiere demasiado. En algunos casos, se tendrá que generalizar en exceso la interfaz componente, provocando que sea más difícil de comprender.

2. Referencias

- “Composite - Patrones estructurales.” *Refactoring.Guru*, https://refactoring.guru/es/design-patterns/composite. Accessed 19 October 2023.
- “Composite Design Pattern.” *SourceMaking*, https://sourcemaking.com/design_patterns/composite. Accessed 19 October 2023.