**Trabajo 2 – Patrones y diseño de software**

**Amilkar Fernando Martínez**

**Cristian Camilo Gutiérrez**

**Erich Alexis Báez**

**Instituto tecnológico metropolitano - ITM.**

**Especialización en ingeniería de software.**

**Patrones y Diseño de Software.**

**Carlos Alejandro Ruiz Ramirez.**

# Definición del problema y aplicación de patrones estructurales.

## Adaptador para integración de sistemas heredados

**Patrón utilizado:** ***Adapter***

**Problema:**En una empresa financiera se requiere integrar un sistema antiguo de gestión de transacciones con una nueva plataforma de software. El sistema heredado ofrece sus servicios a través de métodos que no son compatibles con la nueva arquitectura, por lo tanto, no se puede acceder a sus funcionalidades directamente.

**Solución con el patrón Adapter:**

El patrón Adapter permite que la nueva plataforma interactúe con el sistema antiguo sin modificar su código original. Se implementa una clase adaptadora que transforma las llamadas de la nueva interfaz a las del sistema heredado, facilitando la interoperabilidad entre ambas partes.

## **Proxy para control de acceso a archivos confidenciales**

**Patrón utilizado: *Proxy***

**Problema:**  
En una aplicación de gestión documental, se deben proteger ciertos archivos sensibles. No todos los usuarios tienen los mismos privilegios, y es necesario restringir el acceso a archivos confidenciales dependiendo de si el usuario está autorizado o no.

**Solución con el patrón Proxy:**El patrón *Proxy* actúa como un intermediario que controla el acceso al recurso real (archivo). Antes de permitir operaciones como la lectura, el proxy verifica si el usuario tiene permisos suficientes. De esta forma, se evita el acceso no autorizado sin alterar la lógica del objeto original.

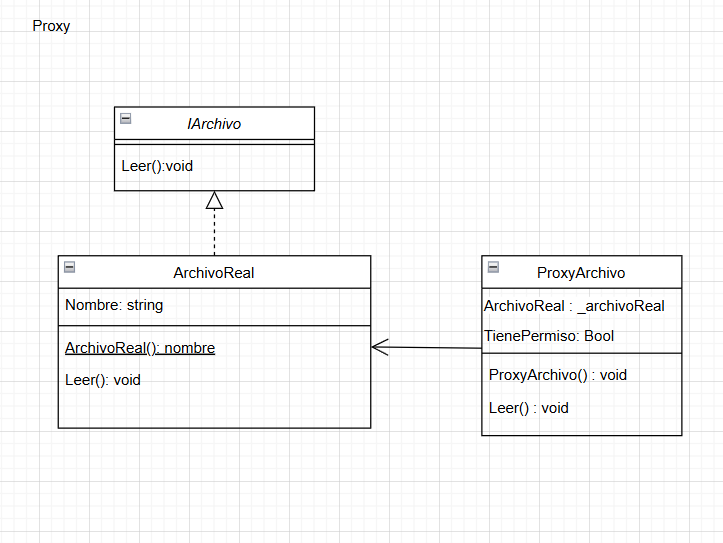
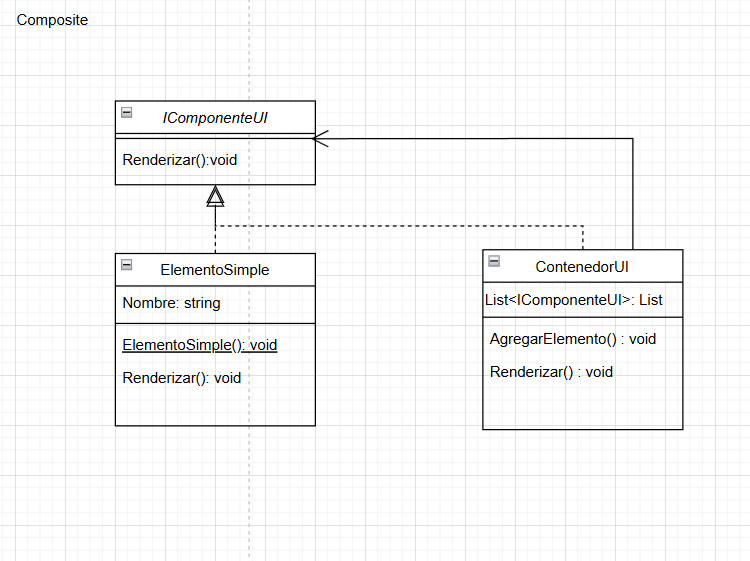
## **Composición flexible de interfaces gráficas**

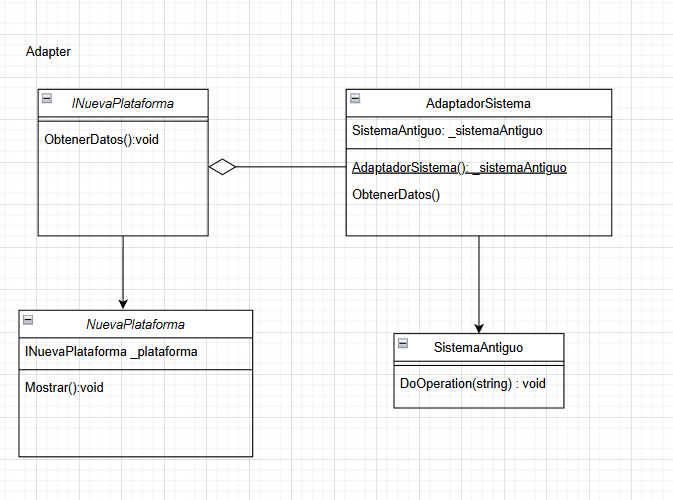
**Patrón utilizado: *Composite***

**Problema:**  
Un sistema de construcción de interfaces gráficas requiere permitir que los formularios sean compuestos de múltiples elementos visuales (botones, etiquetas, paneles), algunos de los cuales también pueden contener otros componentes. La estructura debe ser flexible y escalable.

**Solución con el patrón Composite:**  
El patrón Composite permite tratar de forma uniforme tanto a los elementos simples como a los contenedores que agrupan otros elementos. Esto facilita la construcción jerárquica de interfaces gráficas, permitiendo renderizar componentes anidados de forma recursiva y organizada.

Diagramas:

<https://drive.google.com/file/d/1gtVOW1NxlwXuZVA-Z3cpzxoNgmmU5OES/view?usp=sharing>   
  




Repo: <https://github.com/EbaezrITM/Entregable2.git>