



EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA



TECNOLÓGICO
NACIONAL DE MÉXICO



Instituto Tecnológico de Tijuana
Departamento de Sistemas y Computación

Nombre de la Materia:
Patrones de Diseño

Actividad:
Evaluacion Unidad 3

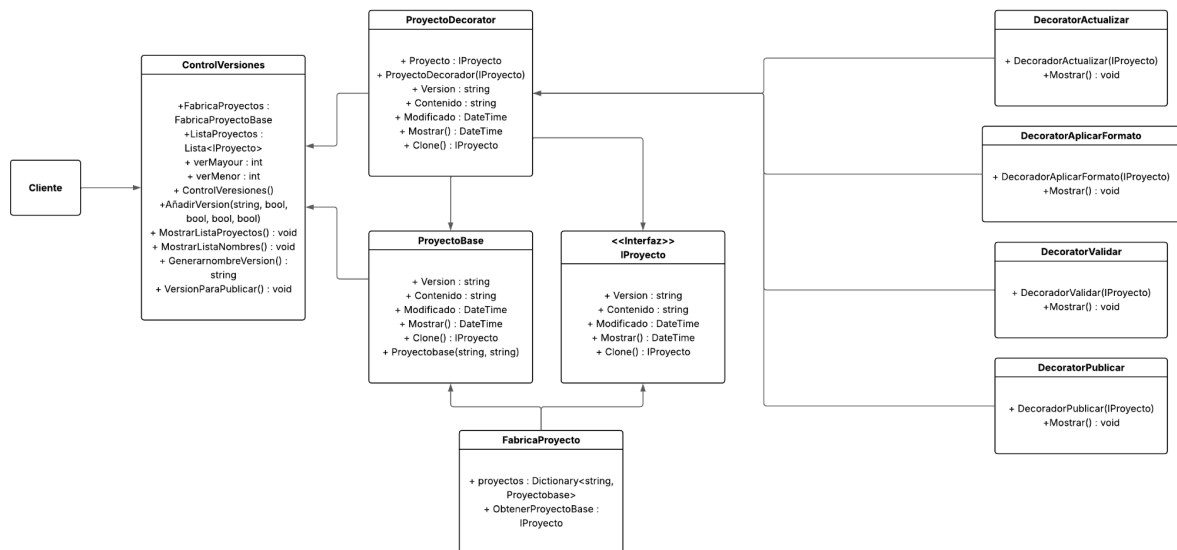
Profesor:
Maribel Guerrero Luis

Alumno(s):
Jonathan Rafael Tamayo Ortiz

Fecha de entrega:
13/11/2025

Evaluacion Unidad 3

Diagrama



Código

```
namespace EvaluacionU3
```

```
{
```

```
    public interface IProyecto
```

```
    {
```

```
        string Version { get; }
```

```
        string Contenido { get; }
```

```
        DateTime Modificado { get; set; }
```

```
        void Mostrar();
```

```
        IProyecto Clone();
```

```
    }
```

```
}
```

```
—
```

```
public class ProyectoBase : IProyecto
```

```
{
```

```
    public string Version { get; }
```

```
    public string Contenido { get; }
```

```
    public DateTime Modificado { get; set; }
```

```
    public ProyectoBase(string version, string contenido)
```

```
    {
```

```
        this.Version = version;
```

```
        this.Contenido = contenido;
```

```
        Modificado = DateTime.Now;
```

```
    }
```

```
    public virtual void Mostrar()
```

```

{
    Console.WriteLine($"\\n\\tVersion: {Version}");
    Console.WriteLine($"\\t\\tUltima modificacion: {Modificado}");
    Console.WriteLine($"\\t\\tContenido:\\n\\t{Contenido}");
}

public virtual IProyecto Clone()
{
    return new ProyectoBase(Version, Contenido)
    {
        Modificado = this.Modificado
    };
}
}

```

```

public class FabricaProyectoBase
{
    private static Dictionary<string, ProyectoBase> proyectos = new Dictionary<string,
ProyectoBase>();

    public IProyecto ObtenerProyectoBase(string version, string contenido)
    {
        if (!proyectos.ContainsKey(version))
        {
            proyectos[version] = new ProyectoBase(version, contenido);
        }
        return proyectos[version].Clone();
    }
    public int TotalProyectosBase => proyectos.Count;
}

```

```

public abstract class ProyectoDecorator : IProyecto
{
    protected IProyecto Proyecto;
    public ProyectoDecorator(IProyecto proyecto)
    {
        Proyecto = proyecto;
        Modificado = DateTime.Now;
    }

    public virtual string Version => Proyecto.Version;
    public virtual string Contenido => Proyecto.Contenido;
    public virtual DateTime Modificado { get; set; }
    public virtual void Mostrar() { Proyecto.Mostrar(); }
    public virtual IProyecto Clone() { return Proyecto.Clone(); }
}

```

```

public class DecoratorActualizar : ProyectoDecorator

```

```

{
    public DecoratorActualizar(IProyecto proyecto) : base(proyecto) { }
    public override void Mostrar()
    {
        base.Mostrar();
        Console.WriteLine("\tAProyecto actualizado.");
    }
}
—
public class DecoratorAplicarFormato : ProyectoDecorator
{
    public DecoratorAplicarFormato(IProyecto proyecto) : base(proyecto) { }
    public override void Mostrar()
    {
        base.Mostrar();
        Console.WriteLine("\tAplicando formato PDF.");
    }
}
—
public class DecoratorPublicar : ProyectoDecorator
{
    public DecoratorPublicar(IProyecto proyecto) : base(proyecto) { }
    public override void Mostrar()
    {
        base.Mostrar();
        Console.WriteLine("\tProyecto publicado.");
    }
}
—
public class DecoratorValidacion : ProyectoDecorator
{
    public DecoratorValidacion(IProyecto proyecto) : base(proyecto) { }
    public override void Mostrar()
    {
        base.Mostrar();
        Console.WriteLine("\tProyecto validado.");
    }
}
—
public class ControlVersiones
{
    private readonly FabricaProyectoBase FabricaProyectos;
    private readonly List<IProyecto> ListaProyectos;
    int verMayor = 0;
    int verMenor = 1;

    public ControlVersiones()
    {
        FabricaProyectos = new FabricaProyectoBase();
        ListaProyectos = new List<IProyecto>();
    }
}

```

```

    }

    public void AñadirVersion(string contenido, bool validar, bool aplicarFormato, bool
publicar, bool actualizar)
    {
        string nombre = GenerarNombreVersion();
        IProyecto proyecto = FabricaProyectos.ObtenerProyectoBase(nombre,
contenido);

        Console.WriteLine($"\\nGenerando version {nombre}...");
        if (actualizar)
            proyecto = new DecoratorActualizar(proyecto);
        if (validar)
            proyecto = new DecoratorValidacion(proyecto);
        if (aplicarFormato)
            proyecto = new DecoratorAplicarFormato(proyecto);
        if (publicar)
            proyecto = new DecoratorPublicar(proyecto);

        ListaProyectos.Add(proyecto);
    }
    public void MostrarListaProyectos()
    {
        Console.WriteLine("Historial de versiones:");

        if (ListaProyectos.Count > 0)
        {
            foreach (var proyecto in ListaProyectos)
            {
                proyecto.Mostrar();
            }
        }
        else
            Console.WriteLine("No hay proyectos en este momento.");
    }

    public void MostrarListaNombres()
    {
        Console.WriteLine("Lista de versiones:");

        if (ListaProyectos.Count > 0)
        {
            foreach (var proyecto in ListaProyectos)
            {
                Console.WriteLine($"{proyecto.Version}");
            }
        }
        else
            Console.WriteLine("No hay proyectos en este momento.");
    }
}

```

```

private string GenerarNombreVersion()
{
    string nombre = $"v{verMayor}.{verMenor}";
    verMenor++;
    return nombre;
}

public void VersionParaPublicar()
{
    verMayor++;
    verMenor = 0;
}
}
—
internal class Program
{
    static void Main(string[] args)
    {
        var controlVersiones = new ControlVersiones();
        bool actualizar, formato, validar, publicar, salir = false;
        string opc = "", seleccion;

        controlVersiones.AñadirVersion("Primera versión del documento", false, false,
false, false);

        do
        {
            actualizar = false; formato = false; validar = false; publicar = false; seleccion
= " ";
            do
            {
                Console.Clear();

                Console.WriteLine("\n=====
=====");
                Console.WriteLine("                MENU PRINCIPAL");

                Console.WriteLine("=====
=====");
                controlVersiones.MostrarListaNombres();

                Console.WriteLine($"
Seleccione todas las opciones que desea
realizar: {seleccion}");
                Console.WriteLine("1. Ver todos los proyectos");
                Console.WriteLine("2. Actualizar proyecto");
                Console.WriteLine("3. Aplicar formato a proyecto");
                Console.WriteLine("4. Validar proyecto");
                Console.WriteLine("5. Publicar proyecto");
                Console.WriteLine("6. Terminar y aplicar la seleccion");

```

```

Console.WriteLine("0. Salir");
Console.Write("Seleccione la opcion a realizar: ");
opc = Console.ReadLine();

switch (opc)
{
    case "1":
        Console.Clear();
        controlVersiones.MostrarListaProyectos();
        Console.WriteLine("\nPresione cualquier boton para continuar...");
        Console.ReadKey();
        break;
    case "2":
        actualizar = true;
        seleccion = seleccion + "2 ";
        break;
    case "3":
        formato = true;
        seleccion = seleccion + "3 ";
        break;
    case "4":
        validar = true;
        seleccion = seleccion + "4 ";
        break;
    case "5":
        publicar = true;
        controlVersiones.VersionParaPublicar();
        seleccion = seleccion + "5 ";
        break;
    case "6":
        controlVersiones.AñadirVersion("Lorem Ipsum", validar, formato,
publicar, actualizar);
        break;

    case "0":
        Console.WriteLine("Saliendo...");
        salir = true;
        break;
    default:
        Console.WriteLine("Esa opcion no es valida.");
        break;
}
} while (opc != "0" && opc != "6");
} while (!salir);
}
}

```

Capturas

```
C:\Users\jonat\source\repos\I  X + v

=====
MENU PRINCIPAL
=====
Lista de versiones:
v0.1

Seleccione todas las opciones que desea realizar: 2 3
1. Ver todos los proyectos

2. Actualizar proyecto
3. Aplicar formato a proyecto
4. Validar proyecto
5. Publicar proyecto
6. Terminar y aplicar la seleccion
0. Salir
Seleccione la opcion a realizar: 6
```

```
C:\Users\jonat\source\repos\I  X + v

Historial de versiones:

Version: v0.1
Ultima modificacion: 11/12/2025 1:04:37 PM
Contenido:
Primera versión del documento

Version: v0.2
Ultima modificacion: 11/12/2025 1:05:00 PM
Contenido:
Lorem Ipsum
Aproyecto actualizado.
Aplicando formato PDF.

Version: v0.3
Ultima modificacion: 11/12/2025 1:05:06 PM
Contenido:
Lorem Ipsum
Proyecto validado.

Presione cualquier boton para continuar...
|
```



```
C:\Users\jonat\source\repos\l  X + v
=====
MENU PRINCIPAL
=====
Lista de versiones:
    v0.1    v0.2    v0.3    v1.0

Seleccione todas las opciones que desea realizar:
1. Ver todos los proyectos

2. Actualizar proyecto
3. Aplicar formato a proyecto
4. Validar proyecto
5. Publicar proyecto
6. Terminar y aplicar la seleccion
0. Salir
Seleccione la opcion a realizar: |
```

Conclusión

El uso simultáneo o de los patrones decorador y flyweight permite la creación de un sistema que usa memoria de forma eficiente, ahorrando espacio de almacenamiento en la creación de múltiples objetos y siendo capaz de diferenciar entre ellos de forma dinámica gracias a los decoradores implementados.